Deep learning for facial emotion recognition

Ruiz-Garcia, Ariel

Award date:
2018

Awarding institution:
Coventry University

Deep Learning for Facial Emotion Recognition

Ariel Ruiz-Garcia

School of Computing, Electronics and Mathematics

Coventry University

A thesis submitted for the degree of

Doctor of Philosophy

September 2018

Certificate of Ethical Approval

Applicant:

Ariel Ruiz Garcia

Project Title:

Deep and Reinforcement Learning for Pose and Illumination Invariant Face and Emotion Recognition

This is to certify that the above named applicant has completed the Coventry University Ethical Approval process and their project has been confirmed and approved as Low Risk

Date of approval:

19 July 2018

Project Reference Number:

P75211

To those who fight...

Dedicado para todos aquellos que luchan, que por alguna razón el progresar parece inalcanzable.
Para todos aquellos que nunca tuvieron la oportunidad...

Acknowledgements

Writing this thesis has been a challenging but exciting and rewarding experience. None of it would have been possible without the support and guidance of my supervisory team. I extend my gratitude to my director of studies, Dr Vasile Palade, for his constant support, for helping me grow as a researcher, and for never saying no to my crazy ideas even if they meant extra work. I extend my gratitude to Dr Mark Elshaw, who has supported me in this journey since day one. I am grateful for his constant encouragement and for always being available to discuss ideas. I would also like to thank Dr Abdulrahman Altahhan and Dr Rahat Iqbal for their support throughout the duration of this project.

Completing this thesis has also been possible thanks to my family, who have supported me unconditionally throughout this journey. I'd like to highlight my gratitude to my parents for believing in me and always supporting my education, and to my sister Kenny who is an inspiration.

I would also like to thank and acknowledge my friends, all of whom in one way or another helped make this thesis possible. I only wish I had enough space to thank each and every one of you, but I'm grateful with everyone. A special thanks to: Ibrahim Almakky, for the many great technical discussions we had, as well as the many ground-breaking (not fake news, believe me) ideas we came up with but never got to implement; Luke Hicks and Erik Barrow who also participated in many great discussions; The PhD students in the Psychology department who helped me stay sane by forcing me to take breaks every now and then (these in particular are a bad influence: Danielle Labhardt, Maria Charalambous, Claire Pillinger); Nicola Webb, who collected some of the data used in this work; Luis Calderon, who played an important role in me making it this far; and all those who proofread this work.

I would also like to thank Coventry University for funding this research. I also thank the founders of the Barry Guidden Grant, as well as the many other bodies that funded my presentations of this work at several international conferences.

Abstract

The ability to perceive and interpret human emotions is an essential aspect of daily life. The recent success of deep learning (DL) has resulted in the ability to utilize automated emotion recognition by classifying affective modalities into a given emotional state. Accordingly, DL has set several state-of-the-art benchmarks on static affective corpora collected in controlled environments. Yet, one of the main limitations of DL based intelligent systems is their inability to generalize on data with nonuniform conditions. For instance, when dealing with images in a real life scenario, where extraneous variables such as natural or artificial lighting are subject to constant change, the resulting changes in the data distribution commonly lead to poor classification performance. These and other constraints, such as: lack of realistic data, changes in facial pose, and high data complexity and dimensionality increase the difficulty of designing DL models for emotion recognition in unconstrained environments.

This thesis investigates the development of deep artificial neural network learning algorithms for emotion recognition with specific attention to illumination and facial pose invariance. Moreover, this research looks at the development of illumination and rotation invariant face detection architectures based on deep reinforcement learning.

The contributions and novelty of this thesis are presented in the form of several deep learning pose and illumination invariant architectures that offer state-of-the-art classification performance on data with nonuniform conditions. Furthermore, a novel deep reinforcement learning architecture for illumination and rotation invariant face detection is also presented. The originality of this work is derived from a variety of novel deep learning paradigms designed for the training of such architectures.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The recent success of deep learning in signal and vision processing related tasks has opened a pathway for the development of intelligent systems capable of reacting to a user's state of mind by recognizing their emotions. Human emotions are an important aspect of every day life and are fundamental for meaningful social interaction. Therefore, it is imperative that automated emotion recognition systems provide good degrees of recognition performance, for instance to avoid misinterpreting a person's state of mind.

This thesis explores the development of novel deep artificial neural network architectures and learning paradigms for emotion recognition from facial expression images. Furthermore, it explores the development of a novel deep reinforcement learning architecture for face detection. The work presented in this thesis takes into consideration the limitations of contemporary state-of-the-art machine learning models designed for face and facial expression recognition and aims to address illumination, face pose, and face rotation invariance, as commonly encountered in real life scenarios.

The face and facial expression recognition architectures proposed in this thesis are also constrained by theoretical aspects of empirical deep and reinforcement learning learning methods. They incorporate and are derived from a variety of concepts, such as transfer learning, domain adaptation, deep convolutional networks, stacked autoencoders, greedy layer-wise unsupervised training, adversarial learning, and deep reinforcement learning, among others.

1.1 Recognizing Human Emotions

Emotion recognition refers to the human ability to perceive and interpret emotions in other people. Recognizing emotions involves analyzing facial expressions, speech signals, hand gestures and other forms of body language, or a combination of these modalities. According to [1], emotions are also essential for social interaction, learning, communication, rational decision-making, perception and cognition. Being able to recognize human emotions is also fundamental for human empathy; when interacting with other people, humans rely on their ability to perceive and interpret emotions in other people and automatically adjust their responses according to the emotional state perceived.

This research focuses on recognition from facial expression images taking into account that it is commonly more feasible to obtain facial images than other sources of affective data, particularly in unconstrained environments. In addition, existing literature shows that recognition from facial expressions can yield higher recognition levels.

1.2 Motivation for Research

The ability to recognize and interpret human emotions is fundamental for meaningful interactions, communication, learning, rational decision-making, perception and cognition. As we continue the transition into a lifestyle that constitutes interacting with intelligent computer systems on a daily basis, it is imperative that these systems possess the ability to react to a user's emotional state and provide appropriate responses that take into account a user's state of mind.

The inspiration for this research is derived from empirical research studies on the importance of emotion recognition during empathy [2]; when empathizing with other people, humans are likely to develop and understanding of other people's emotional

state and unconsciously adjust their responses based on this understanding. For this reason human empathy, and thus human emotions, are often interpreted as an indispensable element of human-human interaction.

There have been many attempts at addressing automated emotion recognition from facial expressions using DL and other machine learning (ML) methods. Accordingly, many state-of-the-art recognition benchmarks have been set on datasets consisting of static facial expression images collected in controlled environments. Yet, when these DL methods are evaluated on images with nonuniform conditions, such as those collected in unconstrained environments, the recognition rate drops dramatically.

Figure 1.1: Sample images from the Multi-PIE dataset illustrating two different levels of relative luminance.

The poor generalization on nonuniform data is partially attributed to the dependency of DL models on large amounts of data, which not always represent the conditions encountered in real life scenarios. Moreover, changes in the data distribution caused by factors, such as changes in illumination, also lead to poor recognition performance. For instance, Figure 1.1 illustrates two images with virtually identical spatial information and different relative luminance levels. Ideally, a DL model should be able to identify these two images as belonging to the same category. However, for a DL model to treat these two images impartially, it requires to see large amounts of data with both conditions during the training phase. This is notably problematic for real life applications intended for use in unconstrained ever-changing environments, where natural and artificial lighting are subject to constant change.

Other forms of variance in the domain of facial expression recognition arise in the form of face pose, rotation, or tilt, all of which significantly affect recognition performance. This is also true in the domain of face recognition where most empirical face

detectors fail to recognize faces that are non-frontal or faces with poor illumination. Theoretically, these generalization and invariance issues can be addressed by training deep networks on very large datasets covering all possible variances. However, the lack of public datasets with realistic conditions, along with the difficulty of training very large DL models, renders the training process virtually unattainable.

Taking into account these limitations of existing DL approaches for emotion recognition, and considering the importance of being able to correctly identify emotions in people, for instance to avoid misinterpretation of a person's state of mind, the work presented in this thesis aims to advance the field of face and emotion recognition in unconstrained environments.

1.3 Scope of Research

This thesis aims to develop novel artificial neural network architectures based on deep and reinforcement learning principles, designed for face and facial expression recognition in unconstrained environments. The research presented in this thesis builds on contemporary theory of empirical learning and optimization paradigms in deep and reinforcement learning with application to emotion recognition and face detection. The overall intended outcome is a set of architectures for face and emotion recognition that work in unconstrained environments.

In this work, the term *emotion recognition* refers to the process of assigning a categorical label to facial expression images using a deep artificial neural network. The categories considered are neutral states and Ekman's Big Six: happy, sad, surprise, angry, disgust, and fear [3]. The latter are commonly considered as universal emotions across cultures and usually develop from a neutral expressions, hence the inclusion of neutral states. This work does not consider other ways to recognize emotions, such as from speech signals or hand gestures and other forms of body language. This is due to the added difficulty of obtaining reliable data in unconstrained environments.

For instance, in a crowded scenario, it is easier to detect faces and facial expressions than it is to detect body language or audio from specific individuals.

The research question addressed in this thesis is as follows:

"Is it possible to develop novel artificial neural network architectures based on deep and reinforcement learning concepts to efficiently recognize faces and human emotions through facial expressions in unconstrained environments?"

In this research question, the phrase *recognition in unconstrained environments* refers to recognition of face and facial expressions under different levels of illumination and facial pose. It also refers to face detection under different levels of face rotation. As a result, this research question is addressed in multiple stages: illumination invariant recognition, facial pose invariant emotion recognition, illumination and rotation invariant face recognition.

1.4 Thesis Originality

The novelty of the research presented in this thesis is in the form of novel deep learning neural network architectures, along with their application to face and emotion recognition from facial expressions, and end-to-end learning algorithms designed specifically to facilitate their training.

Figure 1.2 shows a pictorial summarized description of the illumination and pose invariant emotion and face recognition architectures presented in this thesis. Four different architectures denoted by dotted lines are shown along with their corresponding flow of information. As it can be observed, the overall research question described above is addressed in different stages, each one building upon the previous one, resulting in a framework that addresses facial emotion recognition and face detection in unconstrained environments. More precisely, these architectures address pose and

Figure 1.2: Overview of the emotion recognition models, and face detection model, presented in this thesis. Left to right: Convolutional Ensembles Network, Pose Invariant CNN (pretrained as a GASCA), Illumination Invariant CNN (pretrained as a SCAE), Illumination and Rotation Invariant Q-network (uses SCAE for feature extraction).

illumination invariance in facial emotion recognition, and rotation and illumination invariance face detection.

The contributions presented in Chapters 3, 4, 5 and 6 can be summarized as:

- An illumination invariant Stacked Convolutional Autoencoder (SCAE) model capable of reconstructing images with up to 64 different degrees of illumination as images with the same illumination.
- A Gradual Greedy Layer-Wise (Gradual-GLW) training algorithm that reduces error accumulation in early layers and significantly improves reconstruction performance and training time.
- A pose invariant Generative Adversarial Stacked Convolutional Autoencoder model that can reduce face pose to zero degrees from up to ± 60 degrees.
- Two convolutional layers: one which utilizes $shifting$ neurons, and another one that exploits facial symmetry to reduce its number of parameters.

- Several deep CNN models that achieve state-of-the-art classification rates on data with nonuniform conditions.
- A novel deep reinforcement learning architecture designed for illumination and pose invariant face recognition.

The originality of this work is also derived from a combination of these approaches into a single hybrid architecture for illumination and pose invariant face and facial expression recognition. Other minor contributions include: a derivative of the ReLU transfer function designed to reduce sparsity and constrain image luminance to a given upper and lower bound; deep stacked autoencoder models able to reconstruct as many output planes as produced by convolutional layers in the encoder element; and novel greedy reward policies for deep Q-learning applied to face detection.

1.5 List of Publications

Chapters 2, 3, 4, and 5 contain some excerpts from the following peer reviewed publications that resulted from this research:

Ruiz-Garcia, A., Webb, N., Palade, V., Eastwood, M., & Elshaw, M. (2018). Deep Learning for Real Time Facial Expression Recognition in Social Robots. accepted for publication in International Conference on Neural Information Processing (Vol. 2018December). Siem Reap: Springer.

Ruiz-Garcia, A., Palade, V., Elshaw, M., & Almakky I. (2018). Deep Learning for Illumination Invariant Facial Expression Recognition. In Proceedings of the International Joint Conference on Neural Networks (Vol. 2018September). Rio de Janeiro: IEEE.

Ruiz-Garcia, A., Elshaw, M., Altahhan, A., & Palade, V. (2018). A hybrid deep learning neural approach for emotion recognition from facial expressions for so-

cially assistive robots. Neural Computing and Applications, 29(7), 359373. Springer, https://doi.org/10.1007/s00521-018-3358-8

Ruiz-Garcia, A., Elshaw, M., Altahhan, A., & Palade, V. (2017). Stacked deep convolutional auto-encoders for emotion recognition from facial expressions. In Proceedings of the International Joint Conference on Neural Networks (Vol. 2017May, pp. 15861593). IEEE. https://doi.org/10.1109/IJCNN.2017.7966040

Ruiz-Garcia, A., Elshaw, M., Altahhan, A., & Palade, V. (2016). Deep learning for emotion recognition in faces. In Lecture Notes in Computer Science (Vol. 9887 LNCS, pp. 3846). Springer, https://doi.org/10.1007/978-3-319-44781-0_5

Ruiz-Garcia, A., Elshaw, M., Altahhan, A., & Palade, V. (2016). Emotion Recognition Using Facial Expression Images for a Robotic Companion. In Engineering Applications of Neural Networks: 17th International Conference, EANN 2016, Aberdeen, UK, September 2-5, 2016, Proceedings (pp. 7993). Springer. https://doi.org/10.1007/978-3-319-44188-7_6

1.6 Thesis Overview

The next chapter, Chapter 2, explores existing literature on the nature of human emotions. An in-depth analysis of existing work in the domain of face detection and emotion recognition using deep and reinforcement learning techniques is also provided. The chapter also looks at learning and optimization theory for neural networks and previous attempts to deal with illumination and pose invariance in faces.

In Chapter 3, a new architecture that uses two learning streams to facilitate feature learning is proposed. This chapter also proposes the use of deep convolutional autoencoders to pretrain deep convolutional networks.

Chapter 4 introduces a novel deep learning architecture to address illumination

invariance in facial expression images. This chapter also introduces novel learning concepts that aid in the training of neural networks in general. The method proposed is evaluated on images with very high and extremely low relative luminance levels. The facial expression corpora is also described in detail.

Chapter 5 describes the implementation of several DL architectures that deal with pose invariance in faces. An experimental setup and methodology is proposed and evaluated on multiple datasets. State-of-the-art classification results are reported on several corpora in this chapter. This chapter also combines the learning principles proposed in Chapters 4 and the pose invariant model into a single architecture. This new architecture is evaluated on data collected in unconstrained environments using a NAO robot, a potential application for DL emotion recognition models presented in this thesis.

Since the facial expression algorithms presented in Chapters 4 and 5 are constrained by facial expression images that contain minimal background, and since empirical face detector methods are prone to failure, Chapter 6 proposes a novel deep reinforcement learning architecture designed for face detection in unconstrained environments. It also combines the findings from Chapters 4 and 5 and combines them with deep reinforcement learning principles to achieve good face detection.

Finally, Chapter 7 provides a summary of the findings presented in this work and highlights the novelty of the research presented in this thesis. This is followed by a list of references.

Chapter 2

Background and Literature Review

2.1 Introduction

This research looks at the development of novel face and emotion recognition deep and reinforcement learning architectures as well as learning paradigms. The main objective is to develop DL models for face and emotion recognition from facial expressions, regardless of image illumination and facial pose. Fundamentally, this thesis aims to provide an answer to the research question presented in section 1.3 of Chapter 1.

This chapter provides an overview of artificial neural network learning paradigms. This is followed by an extensive summary of contemporary attempts at automated emotion recognition from facial expressions using DL, as well as automated face recognition using deep reinforcement learning (DRL). Finally, this chapter provides an overview of deep and reinforcement learning paradigms, such as: autoencoders, supervised and unsupervised learning, deep q-learning, transfer learning (TL) and deep convolutional networks, among others, which form the basis of the architectures presented in chapters: 3, 4, 5, and 6.

2.2 Artificial Neural Networks

Artificial neural networks (NN) are computational models inspired by the processing found in the human brain [4]. These sophisticated algorithms are often described as black boxes due to their complex learning process and lack of real explanation for the decisions produced. The most common type are feedforward neural networks, in which information flow only happens in one direction, from input to output. For consistency the term NN refers to feedforward neural networks throughout this work.

In its simplest form, a NN is composed of a single layer with no hidden layers. These are known as single layer perceptrons, and the input vector is directly mapped to the output layer. However, single layer perceptrons can only solve linearly separable problems. Multilayer Perceptron Networks (MLP) [5] are some of the most common classifiers in pattern recognition and overcome the limitation of single layer perceptrons. Learning is commonly done by adjusting the connection weights w between nodes. Nodes are designed to represent a neuron in the human brain and are usually organized in layers which are interconnected. The output for a given node is given by the weighted sum:

$$y = f(\sum_i w_i, x_i + b) \tag{2.1}$$

where f is an activation function, x the input, and b a bias.

Activation functions are commonly employed to provide the network with non-linearity. The most common functions are Sigmoid: $\sigma(x) = \frac{1}{1+e^{-x}}$, Tanh: $tahn(x) = \frac{2}{1+e^{-2x}} + 1$, or rectifier linear unit (ReLU): $y = max(0, x)$ activation functions.

Different variations of NNs have resulted to address specific problem. For instance, recurrent neural networks (RNNs) are designed to deal with sequential data, e.g. temporal or time-series data. Similarly, convolutional neural networks (CNNs) are designed for problems where spatial information is relevant, for instance in visual processing related tasks.

2.3 Deep Learning

According to the Universal Approximation Theorem [6], [7], MLPs with at least one single hidden layer can represent an approximation of any given function. Moreover, the universality of NNs is enabled through the architecture of the NN [6]. However, learning a NN for a given function can be very complex and may require a significant amount of hidden units. In practice, instead of adding more hidden units to the same hidden layer in a NN, it is common to add new layers and allow multiple levels of representation, depending on the complexity of the data from which the model is to learn. This often results in very large models with multiple hidden layers. These large models are referred to as deep NN and are part of a new sub-field within Machine Learning (ML), known as deep learning [8].

Deep learning is concerned with learning data representations and abstractions [9] in a supervised or unsupervised manner. It allows NNs to model complex relationships, whether linear or non-linear, among data. For consistency, in this thesis the term *deep learning* is used to refer to the process of learning data representations with NNs that have more than two hidden layer. These NNs are also referred to as deep NNs. In contrast, NNs with two or less hidden layers are referred to as shallow NNs.

Training of deep NNs is commonly done using backpropagation in conjunction with stochastic gradient decent (SGD). Given a training set x of size N, during training, SGD minimizes the loss:

$$\Theta = \arg\min_{\Theta} \frac{1}{N} \sum_{i=1}^{N} \ell(x_i, \Theta) \tag{2.2}$$

to find the parameters Θ. This is done using mini-batches $x_{1...m}$ of size m. Then the gradients are calculated by:

$$\frac{1}{m} \frac{\partial \ell(x_i, \Theta)}{\partial \Theta} \tag{2.3}$$

where m has to be carefully selected for it to be a good representation of the entire training dataset. One of the main limitations of SGD is that it does not guarantee an

optimal solution, rather just a good local minimum. Nonetheless, the local minimum is often enough.

2.4 Convolutional Neural Networks

This research is concerned with facial expression images and, therefore, employs CNNs taking into account that they have proven to be efficient in visual processing. Convolutional networks [10] are feed forward networks, inspired by the animal cortex, in which nodes are arranged in a two dimensional space in order to take advantage of spatial information. The most common type of CNNs are those applied to two-dimensional data, such as images. As such, the term CNN in this work is used to refer to convolutional networks with two-dimensional filter kernels applied to two-dimensional inputs.

CNNs have the ability to self-learn a vector of salient features, while at the same time retaining spatial information, and, as such offer an outstanding alternative to prescribed feature extraction and representation methods. Moreover, CNNs have significantly fewer parameter than MLPs with the same number of layers, making them less computationally expensive. These are inspired by the receptive fields found in the cat's cortical visual system [11]. Traditionally, every convolutional layer in a CNN often employs more than one filter kernel in order to learn a variety of features that highlight salient information. This results in a set of feature maps; one per filter used. Moreover, because the feature maps are produced by sliding the filter kernel through the image, the information at the edges is often lost. For this reason, it is common to add zero padding, i.e. add zeros around all the edges of the input.

The output of a convolutional layer is defined as the dot product of two matrices AK with m, n dimensions, where A is the current spatial view of a larger matrix I,

and K is the filter kernel. For instance, given $A \in \mathbb{R}^{m \times n}$ and $K \in \mathbb{R}^{m \times n}$ such that:

$$A = \begin{bmatrix} x_{11} & x_{12} & x_{13} & \dots & x_{1n} \\ x_{21} & x_{22} & x_{23} & \dots & x_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{m1} & x_{m2} & x_{m3} & \dots & x_{mn} \end{bmatrix} K = \begin{bmatrix} y_{11} & y_{12} & y_{13} & \dots & y_{1n} \\ y_{21} & y_{22} & y_{23} & \dots & y_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ y_{m1} & y_{m2} & y_{m3} & \dots & y_{mn} \end{bmatrix} \tag{2.4}$$

then the new feature s for the new feature plane at location i, j is given by:

$$s(i,j) = A * K = x_{11}y_{11} + x_{12}y_{12} \cdots + x_{mn}y_{mn} \tag{2.5}$$

This process is repeated over for every location in the image space, for instance, the next value at location $i+1, j$ is produced by $K * A$ but A starts at x_{12} and ends at x_{n+1}. This is only true for weight-sharing convolutional layers where the same filter kernel K is convolved throughout the input image I resulting in a translation invariant feature map. Where equivariance to translation is not needed, e.g. if a the feature of interest is always at the same known location, a filter kernel K can be learned for every spatial view A of the input image I, although this scenario is less common in visual processing.

Traditionally, a CNN is composed of convolution, max pooling, and fully connected layers [12]. Max pooling layers allow the network to down sample the input and speed up training at the cost of giving up some features. The output of Convolutional layers is often shaped by a transfer function. In recent years, most publications employ the rectified linear unit function (ReLU) as this activation function [13]. ReLU layers assist in the training of NN by reducing the risk of vanishing gradients often encountered during training, particularly of very deep NNs. Lastly, Convolutional or Pooling layers are often followed up by an MLP for classification. However, this is not a rule, and the output of a convolutional layer can be directly mapped to an output layer [14].

Finding the right network topology in CNNs is as challenging as it is in traditional MLPs. However, in practice it is common to use small filter kernels. Some of the most commonly used CNN architectures include: residual networks (ResNets) [14], Inception [15], AlexNet [16], and VGG [17]. However, most of these models are very

deep, i.e. have many convolutional layers, and are not necessary for datasets with a small number of classes.

In terms of practical application, CNNs are a popular choice in visual processing related task, particularly in classification. And since they have significantly less number of parameters than MLPs, very deep networks have been employed for large-scale classification [16], [18], [14]. In emotion recognition, CNNs have also set a number of benchmarks on static datasets: [19], [13], [20], [21]. Other work on emotion recognition using CNNs is presented in [22], as well as in [23] —which is a work that resulted from this research —where CNNs are employed for feature extraction and a Support Vector Machine (SVM) [24] is used for classification of the resulting translation invariant feature vector.

2.5 Autoencoders

Autoencoders are neural networks that can reconstruct an input vector and are often used for data dimensionality reduction. They can learn sparse distributed codes similar to those seen in the cortical sensory areas such as visual area V1 [25]. Autoencoders are composed of an encoder function f that learns to map an input distribution $x \in \mathbb{R}^{d_x}$ to a hidden representation $h(x) \in \mathbb{R}^{d_x}$, and a decoder function g that learns to map the hidden representation $h(x)$ back to an approximation $y \in \mathbb{R}^{d_x}$ of the input x. Empirical autoencoders aim to learn nonlinear functions f and g, and constrain h to have a smaller dimensionality than x in order avoid simply learning an identity function and instead learn salient features of the input distribution. This is achieved by minimizing a loss function $L\big(x, g(f(x))\big)$ using empirical training methods such as SGD.

Just like deep NNs, various types of autoencoder variations have been proposed in the literature. In visual processing tasks, the most commonly used variations are: denoising autoencoders [26], which are used to map a corrupted input image to a non-distorted image; variational autoencoders, commonly used to generate images using

random ; sparse autoencoders, which learn sparse representations by having hidden layers larger than the input image; and adversarial autoencoders [27], which rely on adversarial learning as discussed in the next section, and are also used to generate images, often with specific features.

Empirical autoencoders are commonly used as an alternative to dimensionality reduction methods such as PCA, or to pre-train deep neural networks. In contrast, generative class of autoencoders, e.g. variational and adversarial, are commonly used to generate synthetic images. Both approaches are explored in this work as later seen in Chapters 3–6.

2.6 Generative Adversarial Learning

Generative adversarial learning is a relatively new DL framework introduced by [28] and used to train generative adversarial networks (GANs). GANs are composed of two networks: a generative model G and a discriminator model D. Both models are trained simultaneously by playing a min-max adversarial game where the discriminator model tries to determine if a given sample is from the generator or the training dataset. In contrast, the generator maps samples z from a prior distribution $p(z)$ and maps it to the data space. Formally this is defined as:

$$\min_G \max_D E_{x\sim p}[\log D(x)] + E_{z\sim p(z)}[\log(1 - D(G(z)))] \tag{2.6}$$

Although a relatively new sub-field, GANs have become mainstream in synthetic image generation. Accordingly, various works have focused on the generation of realistic synthetic facial expression images. Some of these works include multi-pose face recognition [29], [30], or facial expression image completion [31]. Although GANs are mainly used for data synthesis, some works have explored their use in classification [32].

One of the, arguably major, constrains of GANs is the difficulty in training. GANs are known to be unstable and difficult to optimize. This can be attributed to the large number of parameters to be optimized, as well the joint training process of two networks that have different objectives. However, training deep networks is known to be challenging. Chapter 5 overcomes some of this challenges by combining adversarial learning with an improved version of greedy layer-wise training as described below. The use of GANs for emotion recognition is inspired due to their ability to produce very realistic image reconstructions that retain salient features.

2.7 Regularization

Due to the inherent non-linearity of deep NNs, training can be a difficult task due a several factors, such as: incorrect weight initialization; imprecise network topology, e.g. too many or too few layers, incorrect hyperparameter initialization, e.g. very large or small learning rates, vanishing or exploding gradients; among others.

Several methods have attempted to improve training and generalization of deep NNs, some of which attempt to improve the optimization algorithms directly. For instance, SGD is normally used with momentum. Due to the use of linear activation functions such as sigmoid, training using SGD often leads the network to fall into local minima rather than global minima. This is caused by the significantly small magnitude of the gradients which result in small weight updates, as well as the saturation of gradients by sigmoid activation. Accordingly, momentum aims to overcome this issue by adding a fraction of previous weight updates to the current one. Let $\nabla f(\theta_t)$ be the gradient for the objective function $f(\theta)$ at step θ_t, momentum is given by:

$$\theta_{t+1} = \theta_t + (\mu v_t - \epsilon \nabla f(\theta_t)) \tag{2.7}$$

where ϵ is the learning, μ the momentum coefficient. Similarly, Nesterov momentum aims to improve classical momentum by calculating the gradient at μv_t. Formally, it is given by:

$$\theta_{t+1} = \theta_t + (\mu v_t - \epsilon \nabla f(\theta_t + \mu v_t)) \tag{2.8}$$

Nesterov momentum is known to provide better convergence rates than classical momentum [33], [34]. However, Nesterov momentum is still bounded by some of the constrains of classical momentum; when momentum is too small it cannot avoid local minimum, whereas big momentum may lead to missing the global minimum. An alternative to this is Resilient Backpropagation (Rprop) which addresses these issues by exploiting local gradient information to perform a direct adaptation of the weight step [35].

Although Rprop inherently addresses some of the issues caused by sigmoid activations such as vanishing and saturation of gradients, in practice, ReLU activations have replaced sigmoid functions as the preferred activation function. Moreover, other alternatives to Rprop and SGD have been proposed. For instance, Adam[36] is an alternative to SGD which requires less tuning of hyperparameters, e.g. it computes individual adaptive learning rates. Refer to section 3.3.1 in Chapter 3 for a formal definition of Rprop.

Other regularization methods include the use of dropout [37], which improves generalization performance by preventing the network from co-adapting too much and overfitting. This is done by randomly dropping a set of units and their connection weights during training. Another popular choice is weight decay [38], which attempts to improve generalization by suppressing irrelevant components of the weights vector and the effects caused by static noise on the target. This is achieved by penalizing large weights. Similarly, learning rate decay can assist in improving weight convergence by reducing the size in change.

Another popular choice is Batch Normalization (BN) proposed by [39]. BN is a technique that reduces covariate shift by normalizing the distribution of a each input feature at every layer. Normalization is done by subtracting the batch mean by the batch standard deviation. Moreover, BN reduces the need for other methods such as dropout, and speeds up training by allowing higher learning rates and faster learning rate decays. As presented in [40], we have observed it to also improve generalization performance and therefore it is used throughout this work.

2.8 Greedy Layer-Wise Training

Another way to improve the generalization performance of deep NNs is by pretraining them and using transfer learning. Although random weight initialization is aimed to provide a weight distribution that does not favor any given class, [41] has demonstrated that random weight initialization can lead to convergence in in local minima that are far from an optimal global solution. Greedy layer-wise (GLW) [41] can facilitate the training of deep NNs by treating each individual layer as a shallow network.

In GLW unsupervised training, each individual layer is treated as an individual shallow network and trained individually as an autoencoder. Recall that autoencoders are composed of an encoder function f and a decoder function g. Then, given an unsupervised training function $\mathcal{L}$ which takes as input the training data and returns a trained encoder function $f^{(k)}$, the first layer of the deep NN is trained using raw pixel data. The resulting $f^{(k)}$ is added to a stack of trained encoder functions f. For every remaining layer, pass the raw pixel data through f, and use the resulting features to learn $f^{(k+1)}$ until $k = m$, where m is the number of layers in the deep NN [42]. One all the layers have been trained, one can attach a classification layer to the resulting stack of trained encoders f, and fine-tune for classification using the labels for the data. GLW is exploited in Chapter 3 and improved in Chapter 4.

When there is a lack of data, pre-training using GLW can be done using larger corpora, from different domains. This is particularly relevant for image processing related tasks that employ CNN, given that CNN learn to extract a set of salient features such as shapes, which are commonly found in various domains. This is commonly referred to as transfer learning (TL) and domain adaptation (DA).

2.9 Feature Extraction

Although NNs are powerful function approximators, the high dimensionality of the input data often makes learning difficult. Moreover, high dimensionality often means lengthier training times and increased computational cost. Accordingly, it is common practice to apply a dimensionality reduction procedure such as Principal Component Analysis (PCA) to the training data before learning a model. In emotion recognition from facial expressions, the common approach is to employ Gabor filters [23], [43], [44], [45] to detect edges and highlight salient features. Gabor filters resemble the the perception in the human visual system [43].

Gabor filters are essentially a sinusoidal modulated by a Gaussian kernel function [44] in which orthogonal directions are represented by real and imaginary components. Let λ represent the frequency of the sinusoidal, θn represents the orientation, and σ represents the standard deviation of the Gaussian over x and y dimensions of the sinusoidal plane, the real component of the Gabor filter applied to an image with dimensions the x and y is defined by:

$$\begin{aligned} G_{\lambda,\theta}(x,y) &= \exp\left[-\frac{1}{2}\left\{\frac{x_{\theta n}^2}{\sigma_x^2}+\frac{y_{\theta n}^2}{\sigma_y^2}\right\}\right]\cos(2\pi * \theta n * \lambda)\ . \\ &\text{where} \\ x_{\theta n} &= x(\sin\theta n) + y(\cos\theta n) \\ y_{\theta n} &= x(\cos\theta n) + y(\sin\theta n) \end{aligned} \tag{2.9}$$

The magnitude response after convolving a Gabor filter with over an image is given by:

$$||G_{\lambda,\theta}(x,y)|| = \sqrt{\Re^2\{G_{\lambda,\theta}(x,y)\} + \Im^2\{G_{\lambda,\theta}(x,y)\}}\ . \tag{2.10}$$

where $\Re\{G_{\lambda,\theta}(x,y)$ represents the real part of the filter and $\Im\{G_{\lambda,\theta}(x,y)\}$ represents the imaginary part, as we presented in [23].

Other common pre-processing steps include the use of including Local Binary Pattern (LBP) features [19]. LBP codes are obtained by selecting a group of pixel values, finding the central pixel value and using it as a threshold for each pixel within

the group. Pixel values lower than the threshold value become zero and pixel values above the threshold value become ones. Another popular choice is to employ local transitional pattern (LTP) codes [43]. LTP codes are similar to LBP codes and are obtained by comparing transition of intensity change at different level of neighboring pixels in different direction.

Since faces have specific features, other feature extraction methods exploit these features to extract a set of features. For instance, the work by [46] identifies 15 different feature points and the Euclidean distances between these are used to represent a facial expression. This method requires reconstructing a representation of a neutral face to use as reference.

These feature extraction methods are prescribed and the resulting features are classified using a variety of classifiers such as MLPs or SVMs. SVMs are non-probabilistic binary classifiers well known for per-forming notably well in image classification problems. SVM have also been employed for face recognition problems [47], [48], [19].

2.10 Reinforcement Learning

Reinforcement learning (RL) is an area of research within ML which allows agents to learn from interaction with the environment. RL is usually suitable for problems where there is no known information about the environment, i.e. when there are no labels for the data and no information regarding how the environment will react to a an action taken by the agent.

RL tasks are modeled as finite Markov Decision Processes (MDPs) where an agent can perform a set of actions, A, in a given environment, in order to reach its goal.In such formulation, a learning agent learns by interacting with the environment at given discrete time step, $t \in \mathbb{Z} : t \in 0 \dots m$. At a given time step t, the agent observes the state S_t of the environment and performs an action A_t. Then at time

step $t+1$ the agent receives a reward signal R_{t+1} and is at a new observation S_{t+1} of the environment, both of which are the result of the action it selected at time step t [49]. Formally, the probability of transition from state s to a new state s' after performing an action a is given by:

$$P(s, s') = Pr(s_{t+1} = s' | s_t = s, a_t = a) \tag{2.11}$$

Then the immediate reward signal for the transition (s_t, a_t, s_{t+1}) is $R(s, s')$. During learning, the agent's objective is to reach its desired target while at the same time maximizing the accumulated reward. The cumulative reward is given by:

$$G_t = \sum_{k=0}^{T} R_{t+k+1} \tag{2.12}$$

However, because the way the target is reached is important, in practice it is common to use a discounted reward instead. Discounted reward is used to let the agent know whether short or long term reward is more important. Therefore, it is discounted at every time step t by a factor $\gamma \in [0, 1]$ such that:

$$G_t = \sum_{k=0}^{T} \gamma^k R_{t+k+1} \tag{2.13}$$

Depending on the objective, some variations of RL aim to learn a policy that predicts the maximum expected future reward, e.g. value based RL, model the environment's behaviors, e.g. model based, or learn a policy that defines the agents behavior at a time step t, e.g. policy based. Policy based models have been employed by [50], [51], [52], [53], [54] and [55] for face detection.

2.11 Constrains of State-Of-The-Art Face and Emotion Recognition Models

In the domain of emotion recognition from facial expression images, SVM based methods are a popular choice: [19], [45], [47], [48]. However, these methods are

heavily dependent on image pre-processing methods such as Gabor filters, PCA, or LBP. The main downside to such prescribed methods is the lengthy and difficult process required to craft them, lower generalization performance, latency, among others. As a result, the work in Chapters 3–6 avoids such pre-processing methods and instead rely on deep CNNs for both, feature extraction and classification. The advantages of CNNs over a combination of SVMs and prescribed image pre-processing have also been discussed in our work presented in [23].

The work on emotion recognition using CNNs: [13], [19], [22], [56], [57], also has some limitations. For instance, the method by [57] employs very complex CNN architectures such as inception modules proposed by [18], and does not achieve state-of-the-art performance. Similarly, the work by [22] relies on very large CNNs, namely AlexNet pre-trained on ImageNet [16], which contains over 1.2 million images. Training such large models is likely to require a number of trial and error attempts in order to find the ideal hyperparameters. Moreover, these models rely on very large amounts of data to learn meaningful representations and provide good generalization performance.

The authors of [58] argue that it is imperative to train models with realistic data obtained in the same scenario where the final application will be used. The idea behind this argument is that the models will learn features that generalize the environment, and, therefore, the model will be able to provide better generalization. For instance, one of the main challenges is changes in illumination, which leads to changes in the data distribution. Because most models are trained on static corpora, they fail to adapt to such changes in the data distribution.

Changes in facial pose also lead to changes in the data distribution. Work that has attempted to address pose invariance is computationally expensive and relies on hard-coded features. For instance, the work by [59] requires the use of a template for describing different facial expressions and involves the creation of a model for each person, making it unsuitable for unseen data. Although some work has looked into pose invariant face detection [60], [61], it does not address pose invariant emo-

tion recognition. As a result, Chapter 5 introduces a novel pose invariant emotion recognition model that produces state-of-the-art recognition performance.

Face recognition models are also prone to failure. For instance, as discussed in Chapter 6, empirical methods fail to capture faces on images with low luminance or with some degree of rotation. Other methods relying on RL are also unable to deal with rotation and illumination invariance: [50], [51], [52] and [53].

2.12 Chapter Summary

The aim of this thesis is the development of DL architectures for emotion recognition and face detector. This chapter has introduced deep neural network learning algorithms and optimization methods. A brief introduction on reinforcement learning was also provided. Most of these learning paradigms form the foundation of the work presented throughout this thesis.

This chapter has explored existing sate-of-the-art approaches and highlighted some of their limitations. Some of the main limitations of contemporary approaches to emotion recognition are addressed throughout this thesis: Chapter 3 improves feature learning by exploring ways of improving generalization performance without the need of large corpora. Chapter 4 overcomes illumination invariance. Chapter 5 addresses pose invariance in a much more automated, less computationally expensive, and significantly faster way. Finally, Chapter 6, looks at ways to overcome illumination and rotation invariance on face detection using RL.

Other limitations of training deep NNs are also addressed in this thesis, such as: the difficulty of training GANs, the lack of multi-illumination data, error accumulation problems in GLW training, among others.

Chapter 3

Deep Learning for Emotion Recognition

3.1 Introduction

This chapter considers two deep learning paradigms: transfer learning and convolutional networks, and their application to emotion recognition from facial expression images. The originality of the research presented in this chapter, is in the form of a novel deep CNN architecture with two learning streams to facilitate feature extraction and representation, and the use of deep stacked autoencoders as a pretraining method for deep CNN models that operate in high dimensional feature spaces.

The novel CNN architecture, referred to as Convolutional Ensembles Network (CEN), splits the input image in half according to mouth and eye positions within the image space and feeds each segment to two different ensembles consisting of convolutional layers with several filter kernels. The features learned by both sub-networks are concatenated together before classification is done using a fully connected layer. In contrast, the second CNN architecture proposed is pretrained as a stacked convolutional autoencoder in a greedy layer-wise unsupervised fashion. The SCAE model is capable of learning an approximation of $g(f(x)) = x$ for any number of convolutional layers with high dimensional feature spaces. This preliminary study of stacked AEs also shows that pretraining a CNN in this manner, significantly improves training

time and generalization performance.

The findings presented here serve as foundation for the remaining chapters of this thesis, which rely on deep CNN and TL as the underlying mechanisms for the illumination and pose invariant emotion recognition DL architectures proposed.

3.2 Experimental Setup

3.2.1 Karolinska Directed Emotional Faces Corpus

Figure 3.1: Subject F07 from the KDEF [62] dataset, displaying seven emotions: sad, surprised, neutral, happy, fear, disgust, and angry.

The Karolinska Directed Emotional Faces [62] database (KDEF) is employed to train and test the DL models presented in this chapter. The corpus contains facial expression images belonging to 70 individuals: 35 males and 35 females aged between 20 and 30 years, each displaying seven different emotional expressions from five different angles. All images were taken under a controlled environment, subjects wore uniform T-Shirt colors, and faces were centered with a grid by positioning eyes and mouth in fixed image coordinates [62]. In this chapter only the frontal images, i.e. 0° pose, are considered; a subset containing 140 front angle images for each one of the seven emotions. Refer to Figure 3.1 for a pictorial description.

3.2.2 Image Pre-Processing

In order to facilitate training and to limit unnecessary texture information, dimensionality reduction is applied to all the corpora used in this work by gray-scaling and

resizing the images to 100×100 after extracting the face. Face extraction is done using a distributed version of the detector provided by [63]. The corpus is randomly divided into 70% training and 30% testing subsets. All images are also normalized to zero mean unit variance.

3.3 Deep Convolutional Neural Networks for Emotion Recognition

Unlike traditional feedforward networks like MLPs, CNNs retain spatial information, such as shapes, through filter kernels and therefore are able to identify salient features. In the case of emotion recognition from facial expressions, this is particularly important considering the fact that classification of a given emotion depends predominately upon the shape of facial features such as the eyes, mouth, and eyebrows. However, due to the high complexity of facial expression images, CNN models often require a high number of convolutional layers in order to extract an ideal set of features that best represents the data. A disadvantage of increased network depth is the complexity of the network and training time that grows exponentially with each additional layer. Moreover, increased network complexity often leads to a failure in finding the optimum network configuration, leading to poor generalization performance on unseen data.

This section of the chapter introduces a novel deep CNN, Convolutional Ensembles Network, made up of two ensembles: two sub-networks composed of four convolutional layers each. The main objective is to facilitate learning salient features around the eyes and mouth areas with different parameters, reduce the number of deep learning layers, and therefore simplify the training process.

3.3.1 Convolutional Ensembles Network (CEN)

The two ensembles of the deep CEN model are made up of convolution, ReLU, max pooling, and local response normalization (LRN) layers for feature learning. The resulting translation invariant feature vectors are then concatenated across the first dimension. The concatenation layer is followed by one fully connected layer and one softmaxloss layer for classification. Refer to Figure 3.2 for a pictorial description of this model.

Figure 3.2: Convolutional Ensembles Network.

The convolutional layers retain spatial information through filter kernels and are able to self-extract translation invariant feature vectors of salient features by sharing weights. Their output is defined by:

$$C(x_{u;v}) = (x+a)^n = \sum_{i=-\frac{n}{2}}^{\frac{n}{2}} \sum_{i=-\frac{m}{2}}^{\frac{m}{2}} f_k(i,j)x_u - i, u - j \tag{3.1}$$

where f_k is the filter with a kernel size $n \times m$ applied to the input x [13]. Note that only squared kernels are used throughout this work. The output height, h', and width, w' dimensions produced by convolutional layers are defined by:

$$w' = \lfloor \frac{W + 2 * P_w - k_w}{s_w + 1} \rfloor, h' = \lfloor \frac{H + 2 * P_h - k_h}{s_h + 1} \rfloor \tag{3.2}$$

where H and W denote the height and width dimension of the input image, P denotes the padding across H and W dimensions, and s the stride size.

In addition, the resulting feature planes are uniformly normalized using the Local Response Normalization (LRN) operator [16]. Let k represent the output feature map, and let $G(k) \subset \{1, 2, \ldots, D\}$ represent a corresponding subset of input feature maps, the output of LRN is calculated as follows:

$$y_{ijk} = x_{ijkz} \left(k + \alpha \sum_{t \in G(k)} x_{ijt}^2 \right)^{-\beta} . \tag{3.3}$$

ReLU functions are defined as:

$$y = \max(0, x) \tag{3.4}$$

and facilitate the training of deep models by eliminating the vanishing gradient problem which often renders the training process unsuccessful.

The input is further reduced with max pooling layers. Let x_i be the input and m be the size of the filter, then the output of the max pooling layers is calculated as:

$$M(x_i) = \max \left\{ x_{i+k,i+l} \big| |k| \leq \frac{m}{2}, |l| \leq \frac{m}{2} k, l \in \mathbb{N} \right\} \tag{3.5}$$

The output of the fully connected layer in the CNN is defined according to:

$$F(x) = \sigma(W * x) \tag{3.6}$$

where σ represents a sigmoid activation function defined by:

$$S(x) = \frac{1}{1 + \exp^{-x}} \tag{3.7}$$

As discussed in the literature, Chapter 2, sigmoid activations can lead to vanishing gradients or falling into local minima. As a result, the CEN model is trained using Resilient Backpropagation (Rprop) [35] to avoid such side effects. Let Δ_{ij} represent the individual update-value which determines the size of the weight-update, then the

evolution of the adaptive update-value during learning is based on the error function E according to [35]:

$$\Delta_{ij}^{(t)} = \begin{cases} n^+ * \Delta_{ij}^{(t-1)}, & \text{if } \frac{\partial E}{\partial w_{ij}}^{(t-1)} * \frac{\partial E}{\partial w_{ij}}^{(t)} > 0 \\ n^- * \Delta_{ij}^{(t-1)}, & \text{if } \frac{\partial E}{\partial w_{ij}}^{(t-1)} * \frac{\partial E}{\partial w_{ij}}^{(t)} < 0 \\ \Delta_{ij}^{(t-1)}, & \text{else} \end{cases} . \tag{3.8}$$

$$\text{where } 0 < n^- < 1 < n^+$$

Then the weight-update is defined according to [35]:

$$\Delta_{ij}^{(t)} = \begin{cases} -\Delta w_{ij}^{(t)} & , \quad \text{if } \frac{\partial E}{\partial w_{ij}}^{(t)} > 0 \\ +\Delta w_{ij}^{(t)} & , \quad \text{if } \frac{\partial E}{\partial w_{ij}}^{(t)} < 0 \\ 0 & , \quad \text{else} \end{cases} . \tag{3.9}$$

$$w_{ij}^{(t+1)} = w_{ij}^{(t)} + \Delta w_{ij}^{(t)}$$

This model is trained on the testing subset of the KDEF corpus for $5,280$ epochs. The learning rate for filters and biases was initially set to 1.0 and dynamically adjusted down to 0.00001 over 1000 epochs, whereas the momentum was set to 0.9.

3.3.2 CEN Classification Performance

Table 3.1: Confusion matrix for the CEN model on the test subset of the KDEF corpus. A: angry; D: disgust; F: fear; H: happy; N: neutral; Sa: sad; Su: surprised.

	A	*D*	*F*	*H*	*N*	*Sa*	*Su*
A	95.24	2.38	2.38	0	0	0	0
D	2.38	76.19	2.38	7.14	0	7.14	2.38
F	2.38	4.76	88.10	0	0	0	4.76
H	4.76	0	0	100	0	0	0
N	2.38	4.76	2.38	0	76.19	11.91	2.38
Sa	0	7.14	4.76	2.38	0	85.71	0
Su	0	0	7.14	0	0	0	92.86

The CEN model proposed splits the image horizontally in half, and feeds each half to a corresponding sub architecture to be processed in parallel. Each sub-network

learns a representation of different facial parts: in the case of the first half, the salient features highlighted are the areas around the eyes whereas the second half highlights the area surrounding the mouth. The translation invariant features obtained from each sub-network are then recombined for classification. The CEN model with split input was trained for $5,280$ epochs and achieved an accuracy rate of 86.73%. Table 3.1 illustrates the confusion matrix for this model.

As it can be observed in Table 3.1 the model achieved a higher performance rate when classifying facial images illustrating happy emotions and missclassified neutral faces the most. The misclassification on neutral faces is justified due to the similarity of this emotion with all the others, especially with sadness. As it can be observed in Figure 3.2 above, there is not a big difference between these two expressions and neutral has previously been defined as the basic human emotion [46] which implies that all other emotions evolve from a neutral emotional state.

3.4 Preliminary Evaluation of Stacked Convolutional Autoencoders

Figure 3.3: Illustration of deep CNN model pretrained as a SCAE.

Due to the inherent non-linearity of deep networks, empirical training methods such as SGD may fail if the parameters are not initialized appropriately or if the

network topology is not ideal for the problem being solved, e.g. too many layers or too few convolutional kernels. Imprecise network configurations can lead to large or small gradients and problems in obtaining a set of weights that provide optimal generalization of the training data. Where the topology or parameters of the network are not ideal, it often requires a lengthy training process, particularly for very deep models. Random weight initialization is often the preferred choice among researchers and is intended to provide the network with a weight distribution that does not favor any particular class. However, recent studies [41] show that random initialization of weights can lead to convergence in local minima that are far away from an optimal global solution.

One way to overcome this training difficulty associated with random initialization is by employing autoencoders to pretrain each layer of a CNN in a greedy layer-wise unsupervised manner as discussed in Chapter 2. This allows for an initialization of filter kernels in a CNN close to a good local minimum [41], which leads to improved feature extraction and classification performance. However, empirical CNN models employ a large number of filter kernels and the deeper the layer the more filters used, since this can be afforded computationally, and therefore the increase in dimensionality of the feature vectors. The increase in dimensionality of the feature vector, and other problems such as exploding or vanishing gradients, makes it difficult to train a network to map an input distribution to a hidden representation, and the hidden representation back to an approximation of the input. For this reason, only the first convolutional layer is pretrained as an autoencoder.

This section of the chapter explores whether every layer of a deep CNN can be pretrained in a GLW unsupervised manner, regardless of how large the dimensionality of the feature vector to be reconstructed may be. The proposed stacked convolutional autoencoder (SCAE) utilizes batch normalization (BN) to speed up training using larger learning rates, and ReLU activation functions to avoid vanishing gradients.

3.4.1 Stacked Convolutional Autoencoders

Recall from Chapter 2 that the purpose of an autoencoder is to learn a hidden representation $h(x)$ of the input distribution $x \in \mathbb{R}^{d_x}$. This is achieved by two functions: an encoder and a decoder. Formulating a deep CNN as an autoencoder requires using the original CNN as the encoder element and adding layers to represent the decoder. In order to ensure that the reconstruction y has the same dimensions as x, it is necessary to upsample, or learn a deconvolution procedure, the hidden representation h. This in effect makes learning $f(x)$ and $g(h)$ simultaneously a complex task due to the large number of parameters.

The main challenge with this formulation of the CNN model as an AE is that the number of parameterized layers increases over a magnitude of two, which makes training more difficult. GLW can be employed to gradually learn $f(x)$ and $g(h)$ by deconstructing the autoencoder into smaller shallow autoencoders, consisting of only one parametrised layer in the encoder element and one in the decoder element, and training these individually before combining as a stacked autoencoder.

To build the shallow autoencoders, and eventually the SCAE model, each convolutional layer and its subsequent layers: BN, ReLU, and Max Pooling in some cases, are treated as a single block and the encoder element for each individual autoencoder. An equivalent block of layers which replaces Max Pooling with Upsampling layers is used as the decoder component. Refer to Figure 3.3 for a pictorial representation of the SCAE model and Table 3.2 for a detailed description of the topology. When there is no Max Pooling applied to the output of the convolutional layer, the decoder element does not use an upsampling layer.

The encoder function $f(x)$ of the SCAE model is formally defined as:

$$h = f(x) = s_f\big(Wx + b_h\big) \tag{3.10}$$

where s_f is an activation function, W a weight matrix and b an activation bias. The

decoder function g has the form:

$$y = g(h) = s_g\big(Wh + b_y\big) \tag{3.11}$$

where s_g is the decoder's activation function, a ReLU function in this work and $b_y \in \mathbb{R}^{d_x}$ the bias. Training consists in finding parameters $\theta = W, b_h, b_y$ that minimize the error between reconstructions and inputs over a training set of examples D_n, which corresponds to minimizing the following objective function:

$$JAE(\theta) = \sum_{x \in D_n} L\Big(x, g\big(f(x)\big)\Big) \tag{3.12}$$

where L is a loss function penalizing $g(f(x))$.

Table 3.2: SCAE and CNN topology. Each row, except for the last one, corresponds to an individual AE during GLW training. Final SCAE is obtained by stacking all the encoder layers in the first column, and the decoder layers last column. For the first encoder the initial input is a $1 \times 100 \times 100$ image. The subsequent encoders take as input the hidden representation h from the previous encoder. All Convolutional layers are followed by ReLU and Batch Normalization layers.

CNN/Encoder	Feature Space(h)	Decoder
Convolution $20, 5 \times 5$		Convolution $1, 5 \times 5$
MaxPooling 2×2	b $\times$20a$\times 50 \times 50$	Bipolar Upsampling
Convolution $40, 5 \times 5$		Convolution $20, 5 \times 5$
MaxPooling 2×2	b $\times$40a$\times 26 \times 26$	Bipolar Upsampling
Convolution $60, 3 \times 3$		Convolution $40, 3 \times 3$
MaxPooling 2×2	b $\times 60 \times 14 \times 14$	Bipolar Upsampling
Convolution $80, 3 \times 3$	b $\times 80 \times 14 \times 14$	Convolution $60, 3 \times 3$
MLP	b$\times$100	
SoftMax	b $\times$7	

In the SCAE model, the first autoencoder learns to reconstruct raw pixel data. The second autoencoder learns to reconstruct the output of the first encoder: raw pixel data passed through the first encoder component of the first autoencoder, and so forth. Finally, because the network uses a fully connected layer with 100 hidden units, this layer is trained to associate the output of the last convolutional encoder with its corresponding label. Refer to section 2.8 in Chapter 2 for a detailed description of the training process.

All individual autoencoders are trained for only 10 epochs using mini-batch SGD. Mini-batches are of size 49 and, in the case of the convolutional autoencoders, the loss in Equation 3.12 is measured using the mean absolute value (C) of the element-wise difference between input x and the reconstruction y:

$$C = \frac{\sum_{i=1}^{n} |x_i - y_i|}{n} \tag{3.13}$$

where x and y are both vectors with a total of n elements. In the case of the fully connected layer the loss is measured by the cross-entropy criterion referred:

$$y = -x_c + \log\left(\sum_j \exp(x_j)\right) \tag{3.14}$$

Where there are max pooling layers in the encoder element, these are replaced with nearest neighbor upsampling with a scale of 2. Let u and v represent image coordinates of the input image, α the scale, then upsampling f is defined as:

$$f(u, v) = \lfloor \frac{u-1}{\alpha} \rfloor + 1, \lfloor \frac{v-1}{\alpha} \rfloor + 1 \tag{3.15}$$

Once all the autoencoders are trained, they are stacked together and fine-tuned for reconstruction for 10 epochs. Then the weights corresponding to the encoder layers are used to initialize the CNN model. This CNN is then fine-tuned for classification as a single model for only 20 epochs, also using SGD with a momentum of 0.6 using the criterion described by Equation 3.14. When trained for higher number of epochs

the performance of the network drops or remains the same. Learning rate, LR, for fine-tuning was set to 0.1 and annealed by a factor of 0.001 according to:

$$LR = \frac{\lambda}{1 + (\omega \times \theta)} \tag{3.16}$$

where λ is the initial LR, θ is the decay factor and ω the current epoch.

3.4.2 SCAE and CNN Regression and Classification Results

Figure 3.4: Sample visualization of filter kernels in the first convolutional layer. Left to right, subject F05 of the KDEF dataset illustrating: fear, sad, and happy emotions.

Although GLW training has demonstrated to improve training and generalization of deep autoencoders composed of fully connected layers, MLPs, it is not applied in practice to deep convolutional models. The reason being that empirical deep CNNs utilize a high number of filter kernels, thus producing a multidimensional feature vector h that causes the search space for the decoder function g grow exponentially. Therefore, learning a decoder function g from Equation 3.11, that can accurately produce an approximation of the input x, becomes difficult to accomplish. For this reason, it is common to only pretrain the first layer of a CNN as an AE [64] taking into account than in empirical CNN models the first parametrised layer has the least amount of convolutional kernels and therefore the reconstruction task is much simplified.

One of the main observations made during training was that the first convolutional layer of the SCAE models learns a set of filters that resemble Gabor filters, such as

Table 3.3: Classification performance by the CNN model on the KDEF dataset. A: angry; D: disgust; F: fear; H: happy; N: neutral; Sa: sad; Su: surprised.

	A	*D*	*F*	*H*	*N*	*Sa*	*Su*
A	90.48	2.38	0	0	2.38	4.76	0
D	2.38	90.48	0	0	0	7.14	0
F	2.38	0	83.33	0	0	9.52	4.76
H	0	0	0	97.62	2.38	0	0
N	0	0	2.38	2.38	95.24	0	0
Sa	0	0	4.76	0	0	95.24	0
Su	0	0	2.38	0	0	2.38	95.24

those that we proposed in [45], which are often used for edge detection. Figure 3.4 illustrates sample filter activations of the first convolutional layer when an image labeled as *happy* is forward propagated through the CNN. Notice the main areas highlighted are those around the eyes and mouth, just as it is the case in the work we presented in [45].

When fine-tuning the CNN model for classification using the weights of the encoder element of the SCAE model, the CNN model achieves a classification performance of 92.52% on the test subset of the KDEF dataset. This is an increase of 1.36% compared to when the CNN is not pre-trained as a SCAE but instead is initialized with random weights. Although 1.36% may seem as an insignificant improvement in performance, it is an increase of over 15% on the classification error, which is significant for a classifier. When trained with random weight initialization, the CNN achieved a top classification performance of 91.16% after 500 epochs, compared to a combined of 80 epochs for the CNN: 20 for fine-tuning and 10 for each individual layer including the MLP, and 10 for fine-tuning the entire stack for reconstruction.

As it can be observed in Table 3.3, the CNN emotion recognition model performs well on the emotions *happy*, *neutral*, *sad*, and *surprised* and only misclassifies them once or twice. The worst performance is on the emotion $fear$ which often tends to be confused with other emotions such as *sad*. The misclassification of images belonging to the class fear can be attributed to their similarity to sad images, noticing that sad

images were only confused with fear ones: the shape of facial features, particularly of the eyes and eyebrows tend to be very alike. As illustrated in Figure 3.4, the representations learnt for the sad and fear images are relatively identical, whereas the representation learnt for a happy image is significantly different, particularly in the area around the eyes. In effect, this explains the misclassification of such images and exposes the challenge faced by models intended for real-time emotion recognition: since people express emotions in a number of ways, particularly if ethnic backgrounds are different, it can be difficult to create a model that can efficiently differentiate emotions that are expressed in similar ways.

3.5 Discussion

In this preliminary study of convolutional networks and transfer learning, it has been established that both of these methods are suitable for emotion recognition from facial expressions.

The novel Convolutional Ensembles Network uses two deep learning streams for feature learning. Although the model produces remarkable classification results, it was observed that, due to the use of sigmoid activation functions and random weight initialization, the model tended to fall into bad local minima. This issue was addressed with resilient back propagation. Furthermore, training required several trial and error attempts to find the best initialization parameters. When the parameters were not ideal, it led to exploding or vanishing gradients. The exploding or vanishing gradients problem was addressed in the SCAE model by combined use of ReLU functions along with BN.

It was also established that it is possible to pretrain very deep CNN models—with many filter kernels and high multidimensional feature spaces—as autoencoders in a GLW fashion using empirical learning methods such, as SGD. It was demonstrated that features learned during unsupervised pretraining can be transferred and used in supervised learning. Equally important, it was shown that this approach helps the

CNN model by exponentially reducing its training time and increasing its generalization performance.

One of the main observations during the training of the SCAE model was that the training of the first convolutional layer as an autoencoder largely affects the performance of the remaining layers. Overfitting in the top layer leads to small reconstruction error in the deeper layers when trained individually, however, when the layers are stacked a significant increase in the reconstruction error is observed. This can be explained by the error accumulated in the first layer, which is propagated to deeper layers. The deeper layers are then learning $g(f(x))$, where x is the feature vector produced by the layer before, which may not be a good representation of the original input. In this case, the deeper layers are then learning to reconstruct a feature vector that is far from a good local minima. This problem decreased when BN was used after each convolutional layer. BN also helps in improving training time and avoiding exploding gradients, which was often observed in deeper layers.

The performance achieved by the CNN model pretrained as a SCAE is comparable to more complex DL emotion recognition models with many more parametrised layers [13], [22], [21], and similar performance than models that employ Gabor filters for feature extraction as we presented in [45]. The CNN model proposed in this work self-learns Gabor-like filters with the first convolutional layer and improves the feature vector through lower convolutional layers. Finally, it was also observed that training a SCAE model is challenging not only due to the high number of filters in the deeper convolutional layers, but also due to error accumulated in early layers, which is propagated to deeper layers. This issue is addressed later in Chapter 4.

One of the main differences observed between the CEN model and the CNN model pretrained as a SCAE was the training time it took for each. The CEN model had to be trained for over 5000 epochs compared to 80 for the CNN. Further training of the CEN model also led to overfitting. Although this is partially attributed to the use of BN, the pre-training method proved to be more efficient than random weight initialization and, therefore, this approach is also adopted in Chapters 4, 5 and 6.

3.6 Chapter Conclusion

This chapter has explored two popular concepts in deep learning and their application to emotion recognition. Two main contributions have been proposed: (i) a novel deep convolutional architecture with two learning streams, and (ii) a deep CNN model with high dimensional feature spaces pretrained as a SCAE in a greedy layer-wise unsupervised fashion.

Other contributions are presented in the form of a new insight into the training process of deep CNN with random weight initialization, which leads to vanishing and exploding gradients, or convergence in non-optimal local minima, and the use of resilient back propagation, ReLU activation functions, and batch normalization to address these issues. Similarly, new findings were presented on the use of bipolar upsampling as an alternative to deconvolutional layers, and new knowledge on the features learned by the first convolutional layer in deep SCAE models, which resemble features produced by Gabor filters. In addition to these, new knowledge is presented on unsupervised pretraining using the GLW algorithm: whereas unsupervised learning seems to work and improve the generalization performance of deep CNNs, the GLW method has some shortcomings, such as high error accumulation.

The following chapter addresses one of the main challenges in recognizing emotions through facial expressions, i.e. illumination invariance, and exploits the findings gathered chapter such as the concept of pre-training deep CNNs as SCAE models.

Chapter 4

Illumination Invariant Emotion Recognitinon

4.1 Introduction

When dealing with images or live video feed collected in unconstrained environments, natural and artificial lighting conditions, and therefore image luminance, can drastically change within the span of a few seconds. This is problematic for DL models that are intended for use in real time in ever-changing environments due to changes in the data distribution. Moreover, since it is virtually impossible to obtain data that can accurately represent all possible scenarios, training a NN that can provide a good degree of generalization performance under unforeseen and drastically different conditions remains a challenge in DL.

This chapter of the thesis explores the development of an illumination invariant deep CNN for emotion recognition from faces and builds on the preliminary findings gathered in in Chapter 3 on transfer learning and SCAE models as a means of pre-training deep CNN. The SCAE model presented here learns an internal illumination invariant feature vector h of the data distribution using an improved version of the GLW training algorithm. Two of these models are trained using different corpora in order to provide an in-depth analysis of TL and Domain Adaptation (DA) in the domain of facial expression recognition. The main contributions presented in this chapter are as follows:

- An illumination invariant SCAE model capable of reconstructing images with up to 64 different degrees of illumination as images with virtually the same illumination.
- A Gradual GLW training algorithm that reduces error accumulation in early layers and significantly improves reconstruction performance, training time, and generalization of deep networks.
- An illumination invariant deep CNN emotion recognition model that produces state-of-the-art classification performance on the CK+, JAFFE, FEEDTUM and KDEF corpora.

Other contributions are presented in the form of a derivative of the ReLU activation function with an upper threshold and new insight into how these thresholds affect regression and classification performance. As well as new insight into how γ correction can be used to create a training set when data with varying illumination is unavailable. Furthermore, the use of these learning paradigms in combination with the learning method proposed —using the same image as target for reconstruction for several other images with varying illumination —contribute to the novelty of the work presented in this chapter.

As later discussed in this chapter, when these approaches are combined, an increase in classification accuracy of 5%–15% is observed on different facial expression corpora. Moreover, training time is reduced exponentially, and the SCAE model produces image reconstructions on unseen data with significantly low reconstruction error.

The next section of this chapter introduces the experimental setup and corpora used for this work. The chapter then introduces an illumination invariant CNN model pretrained in an unsupervised fashion as a SCAE using an improved version of the GLW algorithm. The chapter then concludes with a discussion of the findings presented and future work.

4.2 Experimental Setup

Two illumination invariant models are trained in order to illustrate the difference between pretraining on one dataset and fine-tuning on another or pretraining and fine-tuning on the same dataset. The first model, $SCAE_1$ is trained on the Multi-PIE and Yale datasets and evaluated, i.e. the CNN is fine-tuned and tested, on the CK+ and KDEF corpora. The second model, $SCAE_2$ is trained and evaluated on a combined corpus of facial expression datasets, referred to as the Combined Emotional Faces (CEF) dataset hereafter.

4.2.1 Multi-PIE Dataset

The $SCAE_1$ model is trained on the Multi-PIE dataset [65]. This corpus contains a total of $750,000$ images from 337 subjects. These images were collected over four sessions and capture 15 view points and 19 different illumination conditions. For the work described in this chapter, the high resolution images along with the $\pm 90^\circ$ views are discarded and only $580,907$ images covering all 19 illumination conditions and the 13 view points are used. This also excludes images where the face detector failed to capture a face. Note that this corpus has no labels for emotion categories.

4.2.2 Yale Database

The extended Yale Face Database B [66] is also used to train and test the $SCAE_1$ model. This corpus contains a total of 16128 facial images from 28 subjects with 9 poses and 64 degrees of illumination. The corpus also contains an ambient image where all the images for every subject were taken, however these images are not considered in this work. It is worth noting that some of the images in this corpus have relatively low luminance levels making it difficult to visually recognize a face. This corpus also does not have labels for emotion categories.

4.2.3 Facial Expressions Corpora

To evaluate the classification performance of the first illumination invariant emotion recognition model and compare its performance against the models proposed in Chapter 3, it is evaluated on the testing subset of the KDEF corpus. Furthermore, to compare against empirical models, it is also tested on the testing subset of the CK+ corpus.

The second model, $SCAE_2$, is trained on large facial expression database, CEF, consisting of: the CK+; KDEF; Japanese Female Facial Expressions (JAFFE) [67]; and the Facial Expressions and Emotions (FEEDTUM) [68], corpora. The JAFFE dataset consists of 213 images from 10 Japanese female subjects posing seven emotions. The FEEDTUM database contains video streams from 18 participants' reactions to stimuli videos, capturing 7 affective states from neutral to the peak of the emotion. The emotion categories include Ekman's six universal emotions: angry, disgust, fear, happy, neutral, sad, and surprise, plus neutral states. For the FEEDTUM database, the first 30% along with the last 10% of each sequence of images is discarded; since each sequence starts with a neutral face and transitions to an emotion, this ensures that the images used contain the most emotion related information rather than neutral faces.

4.2.4 Image Pre-Processing

Taking into account that color only adds texture information, dimensionality reduction is applied to all the corpora used in this chapter by gray-scaling and resizing the images to 100×100 after extracting the face. Face extraction is done using a distributed version of the detector provided by [63]. For the Multi-PIE dataset, the face detector was executed in each sub-folder —each containing the same image with 19 different degrees of illumination —until the first face was found; since some images are very dark, the face detector fails to find a face. The same bounding box is then used for all the images within the same sub-folder to ensure that the input and target

images for the SCAE contain similar spatial information but different illumination. In contrast, the Yale dataset already provides cropped images containing only the face. Face detection for the CEF corpus is done as explained in Chapter 3. All corpora are randomly divided into 70% training and 30% testing subsets, and all images are also normalized to zero mean unit variance.

To create the training dataset for the SCAE_1 model, the relative luminance, Y, is estimated for every cropped facial image from the Multi-PIE and Yale datasets. In both corpora, each subfolder contains the same facial image with several different illumination conditions: 19 for the Multi-PIE and 64 for the Yale dataset. Therefore, the mean luminance for each subfolder is estimated and the image with luminance level closest to the mean, referred to as x_μ hereafter, becomes the target reconstruction image for all the other images, including itself. This ensures that all images within the same subfolder are reconstructed with the same luminance, regardless of how low or high it is within the original image. Let R, B, G represent the linear red, green and blue, RGB, values of an image before gray-scaling, relative luminance Y for the given image is defined by:

$$Y = 0.2126R + 0.7152G + 0.0722B \tag{4.1}$$

For the second model, SCAE_2, since the CEF corpus does not have images with varying luminance, gamma correction is used to alter image luminance on the training subset. Gamma correction alters the luminance of an image with a non-linear alteration of the input values and the output values. Given an input image i, the gamma corrected image x is defined by:

$$x = \left(\frac{i}{225}\right)^{\frac{1}{\gamma}} \times 225 \tag{4.2}$$

where $\gamma \in \{0.4, 0.6, 0.8, 1.0, \ldots, 3.4\}$.

The use of gamma correction augments the training subset of the CEF dataset over a magnitude of ten. Note that when $\gamma = 1.0$ the input image remains unchanged,

Figure 4.1: Sample images before and after γ corrections.

thus in this case $x = i$. For this reason, when training the SCAE_2 model the gamma corrected image x with $\gamma = 1.0$, also referred to as x_μ for consistency, becomes the target reconstruction image for itself and all other gamma corrected copies of itself. Note that the selected γ values provide a good range of very dark and very light images.

4.3 Illumination Invariant Architecture

Recall the images in Figure 1.1 from Chapter 1. Even though the two images belong to the same subject and contain virtually identical spatial information, in order for a DL model to know that these two images belong to the same subject, or even simply to label them as the same category, will require the model to be trained with enough data that can accurately represent all possible variations of the image, which is often unattainable, particularly when the data is limited. This issue increases the difficulty of being able to recognize emotions from facial expression, particularly in unconstrained environments. This section of the chapter presents the novel SCAE model designed to address illumination invariance.

4.3.1 Unsupervised Feature Learning: Gradual GLW

Due to the inherent non-linearity of deep learning models, empirical training methods such as SGD may fail if the network topology is not ideal for the problem being solved, i.e. too many or too few deep learning layers, too few neurons in MLPs or filter kernels in CNN, or if the network hyperparameters are not properly initialized.

As observed in Chapter 3, these imprecise network configurations can lead to exploding or vanishing gradients, thus rendering the training processing unsuccessful, particularly for very deep models such as autoencoders. Chapter 3 also showed that GLW unsupervised learning of SCAE models can increasingly facilitate the training of very deep CNN models. This section explores this training method further in the context of reconstruction and classification error and looks at ways to overcome the error accumulation observed during GLW training.

As observed in the preliminary experiments, the nature of the GLW training algorithm allows for error accumulated in early layers to be propagated to deeper layers, and therefore deeper layers are often trained to encode or decode features that fall far from a global minimum. This makes it difficult to obtain good image reconstructions y of the input x, even after fine-tuning the final stack of shallow autoencoders for classification. When training a SCAE model for the sole purpose of pretraining a secondary deep model, the ability of the SCAE to produce reconstructions y with significant low reconstruction error is trivial. However, the experimental design of the illumination invariant SCAE requires the model to retain all the spatial information present in the input x. This is also true for other transfer learning or domain adaptation problems.

To address error accumulation, reduce the distance between y and x, improve the overall performance of the GLW training method, and at the same time learn an illumination invariant feature vector, this chapter introduces a novel Gradual Greedy Layer-Wise (Gradual-GLW) training method. Firstly, instead of fine-tuning the final stack only once for classification, the stack of shallow autoencoders is fine-tuned for reconstruction at every step $k \in \mathbb{Z} : k \in \{1, \ldots, m\}$. This inter-layer fine-tuning approach ensures that the shallow autoencoders at steps k and $k+1$ learn to reduce the error accumulated by the shallow autoencoder at step k before the next shallow autoencoder learns to map the hidden representation h produced by these two autoencoders to an approximation y.

Recall that in the preliminary study of SCAE models trained in a GLW manner

the objective was to learn an approximation of $g(f(x)) = x$ by minimizing Equation 3.12. However, when trying to adjust luminance levels on a given image, learning the identify function $g(f(x)) = x$ is not particularly useful given that it only learns to replicate the input image. The objective of the autoencoder model proposed in this chapter is to learn to reconstruct an input image x with relative luminance Y as x_μ, thus the objective is to learn an approximation of $g(f(x)) = x_\mu$, which is achieved by minimizing:

$$JAE(\theta) = \sum_{x \in D_n} L\Big(x_\mu, g\big(f(x)\big)\Big) \tag{4.3}$$

where D_n is the training set, x_μ is an image x with luminance μ and similar spatial information as image x with luminance Y and $\neg\square(\mu = Y)$. Note that $(\neg\square)$ is used to denote that μ and Y are not necessarily equal. Tables 4.1, 4.2 and 4.3 provide a formal definition of the Gradual-GLW training algorithm proposed.

Fine-tuning is done in a similar way as training in Table 4.2 except there are no stopping conditions and is only done for a fixed number of epochs. This is due to the layers already being trained which only require small updates to strengthen the connection between the layers already fine-tuned and the newly trained one.

Minimizing Equation 4.3 can be done using empirical learning methods such as SGD or Adam. Although no significant differences in performance were observed as discussed in the results section, SGD with Nesterov momentum [34] is employed for comparison purposes with the preliminary study discussed in Chapter 3.

Since in previous work Ruiz-Garcia et al. [40] have observed that classifier models with more than four or five convolutional layers do not improve classification performance for the KDEF dataset due to its sparsity, the illumination invariant deep CNN model has five convolutional layers —each convolutional layer becomes a shallow autoencoder in the SCAE model. Additionally, as discussed in the results section, since the data distribution is significantly reduced by the SCAE model, it becomes unnecessary to add more convolutional layers.

Table 4.1: Gradual Greedy Layer-Wise unsupervised training.

Given a training set X and validation set $\tilde{X}$ each containing input images x and target images x_μ, m shallow autoencoders with only two parametrised layers, an unsupervised feature learning algorithm $\mathcal{L}$ —see Table 4.2 —which returns a trained shallow autoencoder, and a fine-tuning algorithm $\mathcal{T}$ —see Table 4.3: train the first shallow autoencoder with raw data and add it to the stack of trained autoencoders f. For the remaining autoencoders: encode the training and validation data using the encoder layers ξ from the stack f and use the resulting features to train the next shallow autoencoder. Add the new autoencoder to the stack and fine-tune the stack on raw pixel data and repeat.
$f^1 \leftarrow \mathcal{L}(f^1, X, \tilde{X})$
$f \leftarrow f \circ f^1$
for $k \leftarrow 2, \ldots, m$ **do**
$[\xi, \delta] \leftarrow f$
$X_f \leftarrow \xi(X)$
$\tilde{X}_f \leftarrow \xi(\tilde{X})$
$f^{(k)} \leftarrow \mathcal{L}(f^{(k)}, X_f, \tilde{X}_f)$
$f \leftarrow f^{(k)} \circ f$
$f \leftarrow \mathcal{T}(f, X, \tilde{X})$
end for
Return f

Table 4.2: Learning procedure for each shallow autoencoder from Table 4.1

Given a training dataset X with m mini-batches of size b, a validation set $\tilde{X}$ and a model f with weight matrix W taking Nin inputs and producing $Nout$ outputs, train f until the difference between the average luminance ι of the reconstructions y and the average luminance μ of the target images is below the threshold Θ or until reaching a maximum number of epochs M. The weight matrix W is initialized with a Xavier distribution [69]. S denotes the interval at which the stopping criteria is assessed.

$V(W) \leftarrow \frac{2}{Nin+Nout}$
for $k = 1, \ldots, M$ **do**
  for $n = 1, \ldots, m$ **do**
    $[x, x_\mu]_n \subset 1, \ldots, m \leftarrow random(X, b)$
    $y \leftarrow predict(x, f)$
    $L \leftarrow loss(x_\mu, y)$
    $f \leftarrow update(f, L)$
  end for
  if $k \bmod S = 0$ ***then***
    $[x, x_\mu] \leftarrow random(\tilde{X}, b)$
    $y \leftarrow predict(x, f)$
    $\iota \leftarrow mean\_luminance(y)$
    $\mu \leftarrow mean\_luminance(x_\mu)$
    if $|\mu - \iota| <= \Theta$ ***then***
      Return f
    end if
  end if
end for
Return f

Table 4.3: Fine-tuning procedure for the stack of autoencoders from Table 4.1

Given a training dataset X with m mini-batches of size b, a validation set $\tilde{X}$ and a model f, train f for M epochs.

for $k = 1, \ldots, M$ **do**
  for $n = 1, \ldots, m$ **do**
    $[x, x_\mu]_n \subset 1, \ldots, m \leftarrow random(X, b)$
    $y \leftarrow predict(x, f)$
    $L \leftarrow loss(x_\mu, y)$
    $f \leftarrow update(f, L)$
  end for
end for
Return f

One of the main challenges in unsupervised learning is determining when to stop training; since there are no labels, there is no direct way to measure the model's performance. Empirically, training is stopped when the error stops decreasing for a given number of iterations. However, in the case of the SCAE model proposed here, it is imperative to avoid overtraining and converging to a model that has learnt an identity function $g(f(x)) = x$ instead of $g(f(x)) = x_\mu$, which in effect would mean $f(x)$ does not result in an illumination invariant feature vector h. Moreover, because the error is estimated according to the distance between the reconstructed image y and the target image x_μ, the error does not necessarily reflect the model's ability to produce an illumination invariant feature vector given that y is an approximation of x and $\neg\Box(x = x_\mu)$. Therefore, the stopping criteria is based on the luminance level of the reconstructed images as illustrated in Table 4.2.

Since error accumulation is not an issue using the Gradual-GLW training method as opposed to the GLW method, each shallow autoencoder in the SCAE_1 model is trained and fine-tuned for only two epochs, compared to ten in the preliminary study. Similarly, the shallow autoencoders in the SCAE_2 model are trained and fine-tuned for only one epoch. Note that because the corpora used to train both models are significantly larger, training using GLW over Gradual-GLW would require training

for much longer.

Each CNN model is formulated as a SCAE as discussed in section 3.4.1, i.e. each parametrised layer is used as the encoder element and a decoder is created using the same layers with upsampling replacing max pooling. Although deconvolutional layers seem a perfect fit for this purpose, nearest neighbor upsampling produces significantly smoother reconstructions, and it facilitates evaluating the luminance of the reconstructions produced by the SCAE models. The reconstruction loss is measured for mini-batches of size 512 using the mean absolute value C from Equation 3.13. Other learning parameters such as momentum and LR decay remained the same as in the preliminary study, 0.6 and 0.001. Due to a significant reduction in error reconstructions using the Gradual-GLW approach proposed in this thesis, higher LRs can be used for deeper layers which in effect allows for faster training. Therefore, $\lambda \in \{0.1, 0.3, 0.5, 0.7, 0.75\}$ were used as initial learning rate for k shallow autoencoders. The same hyperparameters were used for SCAE_1 and SCAE_2.

4.3.2 Classification: Convolutional Neural Networks

Once the SCAE models are trained to learn a feature vector that is illumination invariant, the decoder is discarded and replaced with two fully connected layers of size 5000 and 1000. The output of the first fully connected layer is shaped using a standard ReLU layer, whereas the second is shaped by a ReLU-n layer. The CNNs are fine-tuned for classification using SGD. The output of the CNN model is defined by a SoftMax operator and the cross-entropy loss y as defined by Equation 3.14.

Note that the the fully connected layers do not use batch normalization. Additionally, the first two convolutional layers use a 5×5 kernels and the remaining layers use 3×3 kernels. This ensures that emphasis is placed on smaller shapes, which for the purpose of this work helps identify small salient features that differentiate emotions.

The encoder element of the $SCAE_1$ is used to initialize two convolutional networks, namely CNN_{1a} and CNN_{1b}. The former is fine-tuned and tested on the KDEF dataset and the latter on the CK+ dataset. $SCAE_2$ is used to initialize a third model, CNN_2 which is fine-tuned and tested on the CEF corpus.

Fine-tuning for the CNN_{1b} is done using mini-batches of size 49 and the training subset of the KDEF dataset. The n value for the activation function ReLU was set to 0.4., whereas the learning rate was initially set to 0.1 and annealed by a factor of 0.1 according to Equation 3.16.

For the CNN_{1b}, fine-tuning is done on the training subset of the CK+ corpus using mini-batches of size 38, a learning rate of 0.3 which is annealed by a factor of 0.01. As discussed in the results section, the ReLU-n function in the last fully connected layer was modified with an n value equals to π, and the lower bound was set to 0.1 instead of 0.

Lastly, fine-tuning and testing CNN_2 on the CEF corpus is using the same hyperparameters used for CNN_{1b}, with the exception of batch size which was set to 64. Momentum was set to 0.7 for all three CNNs and all three models were fine-tuned for 10 epochs. Further training did not provide an increase in classification performance.

4.3.3 Weight Activations and ReLU-n

While training using the Gradual-GLW training method, it was observed that reconstructions for images with very low luminance had a marginally smaller reconstruction error than those for images with high luminance. This observation is justified by the use of ReLU transfer functions, which constrain the output of a convolutional layer to non-negative real values $\mathbb{R}_{\geq 0}$, therefore, dark pixel values which are close to zero and often become negative when forward propagated through the network, are bound to remain non-negative. And because there is no upper threshold in ReLU functions the bright pixels —there is a tendency for larger values to continue growing and smaller

ones to become smaller —can grow endlessly. Moreover, when back tracing individual activations, it was observed that pixel values with very high white intensities tend to become large without the the ReLU-n layers used in the SCAE model. This is particularly relevant for the images with very high luminance, which are often the ones misclassified.

When using ReLU-6 activation functions, i.e. imposing a max upper threshold value of 6 as opposed to no upper threshold in ReLU functions [70], the reconstructions for images with high luminance improved marginally. Just as is the case with images with relatively low luminance, by bounding the gradients to remain small, when the image is propagated through the network, the brighter pixel values are not allowed to become too large and the luminance levels of the reconstruction are somewhat controlled. Note that there is a tendency for bright pixels to cause large activations in the network.

This upper threshold also assists in avoiding the exploding gradients problem: Similarly to the vanishing gradient problem, exploding gradients are a common issue when training deep models and are often caused by imperfect network configurations or incorrect parameter initialization, causing the gradients to grow exponentially and eventually rendering the training process a failure. However, when the gradients are restricted to a max value of 6, the exploding gradient problem is reduced and the neural network is forced to learn less sparse representations.

These observations raise the question whether these thresholds are optimal to address illumination invariance. Further experimentation established that a max value of 1 led to faster learning of the SCAE model. Let n represent the upper threshold, the output y of the ReLU-n proposed is defined by:

$$y = \min(\max(x, 0), n) \tag{4.4}$$

Using this upper threshold encourages the network to learn even more sparse features in earlier layers and encourages the network to increase or decrease luminance on the input image, without shifting towards one end in the SCAE models. However,

it was observed that for the fully connected layers in the CNN models, the higher the n values, the larger the decrease in classification performance of the model. After more experimentation it was established that for the fully connected layers, which essentially are responsible for classifying the features learned by the convolutional layers, the upper threshold could be smaller than 1. For the KDEF corpus, it was found that an n value between 0.4 and 1 provided better results on the test set. Similarly, for the CK+ corpus, it was found that a value as small as π provided better results.

The difference in n values for the two corpora is hypothesized to be due to the different in relative luminance between the datasets: the CK+ has a higher mean luminance value due to many images being significantly bright. And, as observed in the SCAE models, the brighter pixel values tend to become very large. Moreover, because the classification on facial expression images depends upon salient features such as the mouth, eyes and eyebrows, with a small upper threshold all the white noise is ignored.

Another issue observed was that when a large amount of activations in the fully connected layers fall below zero during fine-tuning —note that this is not the case for the convolutional layers given they have been pretrained with a lower threshold of zero and the fully connected layers are initialized with a random distribution—the layers struggle to learn. This highlighted an unnoticed issue with ReLU layers and seemed a common problem when fine-tuning on the CK+ corpus. Therefore, to avoid zero multiplications, the lower bound of the ReLU-n functions was set to 0.1 in the last fully connected layer for the CNN_{1a} model. These configurations produced the best results for the CK+ and KDEF corpora, as discussed in the results section. CNN_2 used the same values as CNN_{1b} fine-tuned on the KDEF corpus.

4.4 Results

Training a deep learning model to deal with illumination is a challenging task due to a number of factors such as limited multi-illumination training data, or the large distribution of data containing different illumination variations, which causes the search space to grow exponentially. These issues are addressed in this chapter by employing gamma γ correction to augment a dataset and obtain images with varying luminance, and by training an autoencoder to reduce the data distribution and thus the search space. The data distribution is reduced by learning to encode a set of images containing identical spatial information, but varying illumination as an illumination invariant downsampled feature vector.

4.4.1 Illumination Invariant Reconstruction Results

Figure 4.2: Performance comparison of $SCAE_1$ on unseen data, (left images), when trained using the Gradual-GLW method, (middle images), proposed in this chapter, versus the empirical GLW method, (right images), as used in the preliminary study in Chapter 3. Input images extracted from the CK+, (left), and KDEF, (right), corpora.

The SCAE models are trained using an improved version of the GLW algorithm, namely Gradual-GLW, and learn to produce remarkable reconstructions even on unseen data from different datasets. As it can be observed in Figure 4.2, the SCAE model learns to increase relative luminance on images with low luminance - left image, or reduce the luminance for images with relatively high luminance - right image. Notice that the reconstructions produced with Greedy-GLW retain all the spatial information, as opposed to those produced with GLW.

Ideally, due to the ability of convolutional networks to retain spatial information

through filter kernels, the SCAE models should be able to produce reconstructions that resemble the input image. However, as it observed in Figure 4.2, it is not the case when the SCAE models are trained in a GLW unsupervised fashion. This is due to the error accumulation problem that leads deeper layers to reconstruct a feature vector that falls far form a good local minimum. And once the weights have shifted in a given direction, it is difficult to adjust them in such a way that would allow them produce better reconstructions.

Figure 4.3: SCAE_1 sample reconstructions on the test subset of the Yale corpus. Top row: the input image x; and bottom row: the corresponding reconstruction y.

The Gradual-GLW training method proposed here overcomes the limitations of the empirical GLW training method and significantly reduces training time and reconstruction error. As a result, SCAE models also improve their generalization abilities and are able to produce remarkable illumination invariant reconstructions even on unseen data. When evaluated on the same dataset, the reconstructions are more remarkable and are difficult to differentiate from the ground truth images as observed in Figure 4.3.

As it can be observed in Figure 4.3, even when the input images are significantly dark, i.e. have very low relative luminance levels, and half of the face is not clearly visible, the SCAE model compensates for missing information and produces images not much different than the target, supporting the superiority of the Gradual-GLW training method over the empirical GLW method proposed by [41]. The main advantage of this is that all the spatial information and salient features necessary for classification are kept almost intact. Equally important, when the input image already has a good degree of illumination, this is kept unchanged as observed in the last column of Figure 4.3.

4.4.2 Classification Results

Once the SCAE models converged to a good local minimum that allowed to to produce illumination invariant reconstructions, the encoder element which produces an illumination invariant feature vector h is used to initialize a deep CNN model. Two CNN models share the same autoencoder SCAE_1 except they are fine-tuned on different datasets. CNN_{1a} is fine-tuned on the training subset of the CK+ corpus and achieves a classification performance of 94.90%. CNN_{1b}, fine-tuned on the KDEF, achieves a state-of-the-art classification rate of 95.70% on the testing subset. These results are also reported in [71], which is published work that resulted from this research.

Table 4.4: Left: Classification performance of the CNN_{1a} model on the CK+ (93 out of 98 images correctly classified 94.90%). Right: Classification performance of the CNN_{1b} model on the KDEF (281 out of 294 images correctly classified 95.70%). A: angry; D: disgust; F: fear; H: happy; N: neutral; Sa: sad; Su: surprised.

	A	D	F	H	N	Sa	Su	A	D	F	H	N	Sa	Su
A	76.92	0	0	0	0	23	0	95.24	4.76	0	0	0	0	0
D		100	0	0	0	0	0	2.38	95.24	0	0	0	2.38	0
F	0	0	85.71	0	14.26	0	0	0	0	90.48	0	2.38	2.38	4.76
H	0	0	0	100	0	0	0	0	0	0	97.62	2.38	0	0
N	0	0	0	0	100	0	0	0	0	0	0	100	0	0
Sa	0	0	0	0	16.66	83.33	0	0	2.38	0	0	7.14	90.48	0
Su	0	0	0	0	0	0	100	0	0	0	0	0	0	100

The difference in performance between SCAE_{1a} and SCAE_{1b}, which was observed to affected by the upper and lower bounds of the ReLU-n function, can be justified by the different complexity of each dataset and the larger number of samples in the KDEF. With these observations made, it is possible to conclude that restricting the output of the classifier layer to values between 0 and 0.4 provides a similar effect to dropout [37], by dropping high or low neuron activations. Nonetheless, in this case, the values being dropped are those that have become too small or two large, instead of random ones. Keeping the weights of the convolutional layers, which were pretrained as a SCAE, fixed during fine-tuning, also produced lower classification performance. This can be justified by the fact that the SCAE model only learns to map the input image to an approximate reconstruction and does not take into account categorical

information. Nonetheless, this needs to be explored further. Other configurations also produced lower performance, for instance, adding normalizing the output of the fully connected layers using BN also reduced the performance. In addition to this, when training the SCAE to reconstruct the input image as the image with the highest luminance level instead of the images closest to the mean, x_μ, it was observed that even though the reconstructions for the Multi-PIE dataset were visually remarkable and with a luminance relatively close to that of x_μ, the classification performance dropped.

The CNN_2 model, which is initialized with the weights of the encoder element of the SCAE_2 model and fine-tuned on the testing subset of the CEF dataset, produces a state-of-the-art classification rate of 99.14% on the test subset. Note that the testing subsets of the CK+ and KDEF corpora from Table 4.4 form part of the CEF testing subset. As it can be observed, the CNN_2 model outperforms both CNN_{1a} and CNN_{1b} models, supporting the potential of the Gradual-GLW method in combination with gamma γ corrected images when lack of multi-illumination data is present.

Table 4.5: Classification performance (99.14%) on the CFE corpus, composed of the CK+, KDEF, JAFFE, and FEEDTUM corpora combined.

	A	D	F	H	N	Sa	Su
A	99.34	0.53	0 .13	0.00	0.00	0.00	00.0
D	0.18	99.18	0.00	0.18	0.00	0.36	0.09
F	0.18	0.00	99.04	0.09	0.09	0.35	0.17
H	0.26	0.00	0.11	99.31	0.11	0.34	0.11
N	0.00	0.00	0.00	0.40	97.21	2.79	0 .00
Sa	0.80	0.06	0.25	0.00	0.26	99.05	0.06
Su	0.32	0.00	0.12	0.25	0.00	0.00	99.63

As observed in Table 4.4, the classification performance of the illumination invariant CNN_{1a} and CNN_{1b} models is consistent on both datasets. Both models obtain lower classification rates on *angry*, $fear$, and *sad*. And even though CNN_{1a} achieves 100% accuracy on four out of seven classes, its performance on the most missclassified classes is far worse than the performance of CNN_{1b}, hence the overall lower performance. In contrast, CNN_2 learns to improve the classification performance on

these particular classes but fails to provide the same level of accuracy on *neutral* states. This can be justified by the incorporation of the FEEDTUM dataset in the CEF corpus: although the first 30% of every sequence is discarded along with the last 10%, this does not guarantee that all the neutral faces are removed from every sequence, resulting in many neutral faces being mislabeled as other emotions.

Since the $SCAE_1$ model was trained on a significantly larger dataset and produces remarkable illumination invariant reconstructions, theoretically, the classifiers initialized with this model should yield higher classification rates than the one pretrained with $SCAE_2$ However, this is not the case. The significant increase in performance offered by the CNN_2 model can be justified by the fine-tuning process, which used more facial expressions data with varying conditions than CNN_{1a} and CNN_{1b}. With these observations, it is possible to conclude that better reconstructions do not necessarily mean better classification, and it highlights the importance of fine-tuning on large amounts of data. At the same time, it can be concluded that the use of gamma γ correction in conjunction with Gradual-GLW can yield state-of-the-art classification performance. Furthermore, when unsupervised pretraining is done using the empirical GLW method as is, and ReLU activations instead of ReLU-n as done in Chapter 3, the best performance obtained on the CK+ is of 86% and 91.5% on the KDEF. This difference in classification performance also supports the training method presented in this chapter, even when luminance is not a direct issue.

4.5 Comparison Against State-Of-The-Art

Contemporary attempts to address illumination invariance in the domain of facial expression recognition include the use of noise injection [72], blurring images with Gaussian filters [73], a combination of histograms, principal component analysis (PCA) and discrete cosine transforms [74], or complex and very deep CNN architectures [75]. Although some of these methods produce remarkable results, they are still unable to generalize on data with nonuniform distributions, particularly the hand-crafted methods such as Gaussian filters or histograms. The DL based models are also prone to

overfitting. The novel DL architecture presented here is capable of dealing with data with nonuniform conditions, and can deal with up to 64 degrees of illumination. This architecture presented here extends the findings gathered in the preliminary experiment discussed in Chapter 3 by improving the performance of GLW and proposing a novel training method to address illumination invariance.

The results obtained on the CK+ corpus significantly surpass similar work designed to address illumination invariance [76] and are in line with the results obtained by [73], who use a much more complex approach. [73] use spatial-temporal and universal manifold models to extract low-level features and construction expressionlets. This approach is similar to Action Units [77], and the authors report an accuracy rate of 95.1% on the CK+ dataset. Similarly, [76] use an adaptive filer based on temporal local scale normalization, and use a complex architecture based on a very deep CNN followed by fully connected layers and deconvolutional layers to learn seven different facial expressions from short video clips. This architecture is able to reconstruct the input image, like an autoencoder, as well as categorizing it. The authors report a performance rate of 83% on the CK+ dataset. On the CK+ dataset, the CNN_{1a} model learned to classify four classes out of seven with 100% accuracy, that is two classes more with 100% accuracy than the approach proposed by [76], and three more than the work by [73]. The lowest performance was on angry, which is confused with sad; this marginally surpasses the results by [76] and falls behind on those obtained by [73] for this particular class. Both [76] and [73] obtain 50% or lower on sad, compared to 83.3% using our approach. Nevertheless, the CNN_2 model outperforms all these different approaches with a state-of-the-art classification accuracy of 99.14%.

Compared to empirical DL models, such as the deep CNN used in the preliminary study, the illumination invariant model here learns to exponentially reduce the search space by learning a feature vector that is illumination invariant. For instance, the Yale dataset has sixty-four estimated different levels of illumination. A traditional CNN model would have to learn at every layer that these sixty-four images belong to the same subject and same category. This is particularly problematic for the classification layer, given it has to learn to categorize all these variation under the same category.

The classification layer of the illumination invariant model presented here only sees one level of illumination, and thus it does not need to learn that two or more images with significantly different luminance levels fall under the same category or contain the same spatial information.

Other advantages of the unsupervised pretraining approach proposed in this chapter over contemporary methods is the faster training times allowed by Gradual-GLW algorithm, as well as reduced need for very deep models and increased network complexity. Moreover, the illumination invariance models also have the potential of being employed for other visual processing tasks and should reduce the need for hard coded image pre-processing approaches such as those employed to train very deep networks such as ResNets, which rely on pre-processing techniques such as: alteration of brightness, contrast, saturation, color normalization, and PCA based lighting. In addition, theoretically, this illumination invariant unsupervised training of autoencoders should reduce the need for more complex methods such as denoising autoencoders, which inject random noise to the input images during run time, and should eliminate the need for other techniques, such as dropout, by employing ReLU-n activation functions, as seen in section 4.3.3

For the SCAE models, it is difficult to compare their performance in terms of reconstruction due to a lack of existing work exploring this particular issue, specifically in terms of illumination invariant facial expression recognition. Nonetheless, the regression results presented in this chapter can be used as a benchmark for future work. Note that although this method was only evaluated on gray-scaled images, it can also be evaluated with colored images.

4.6 Chapter Conclusion

This chapter of the thesis has presented a novel deep learning approach to deal with illumination invariance in images with application to facial expression recognition. The approach presented employs a deep CNN pretrained as a SCAE model that

learns to map an input image x to a hidden illumination invariant representation h and back to an illumination invariant approximation y of the input image. The SCAE model is trained using Gradual-GLW, an improved version of GLW also proposed in this chapter, that reduces error accumulation in early layers and significantly improves training time and generalization performance. The encoder element, which produces h, is used to initialize the CNN model, which produces state-of-the-art classification performance. The CNN offers an increase of over 15% in classification performance compared to contemporary methods also designed for illumination invariant emotion recognition from facial expressions, and up to 10% increase when compared to a similar approach that employs GLW as opposed to Gradual-GLW.

The originality of the illumination invariant architecture relies on the unsupervised pretraining approach presented, which learns to increase or decrease illumination in images or keep it the same if it is already good enough —i.e., if salient facial features are already visible. This method also compensates for missing information and is able to reconstruct faces in which some features are not visible due to poor illumination. Moreover, this method provides remarkable generalization performance and is able to produce illumination invariant reconstructions even on unseen data from different corpora. Although the method presented here relies on multi-illumination corpora to learn, it was demonstrated that when there exists lack of multi-illumination data, γ correction can be utilized to magnify the training data.

The work presented here brings us a step closer to emotion recognition in unconstrained environments with non-uniform illumination conditions. However, the main limitation of this work is that it only addresses illumination invariance and does not deal with another important problem in the field of face and facial expression recognition: pose invariance. The following chapter will explore this problem in detail and introduce a novel method to address pose invariance.

Chapter 5

Pose Invariant Emotion Recognition

5.1 Introduction

One of the main findings presented in Chapter 4 was that we can learn an encoder function f that maps an input image x to a hidden illumination invariant representation $h = f(x)$, and learn a function g that maps h to a reconstruction $y = g(f(x))$ that resembles a desired target x_μ and $\neg\Box(x = x_\mu)$. In theory, this establishes that the input and target vectors in an autoencoder do not need to be the same, and therefore we can learn a function that maps an input from a given distribution to a target that lies in a different distribution. This chapter exploits these observations along with Gradual-GLW training and contemporary adversarial learning principles to address pose invariance in the domain of facial expression emotion recognition. A novel convolutional layer with *shifting neurons* is introduced as part of a new architecture that gradually learns to shift faces with a pose of up to 60 degrees to a representation of the same faces at 0 degrees. The resulting latent feature vector representing the input image at 0 degrees is then used for classification. The main contributions presented in this chapter are:

- a novel deep Generative Adversarial Stacked Convolutional Autoencoder (GASCA) model that learns to shift faces with facial pose of up 60 degrees to 0 degrees

representations.

- a hybrid deep learning layer employing convolutional filters to retain spatial information and learn salient features, and fully connected units shared across the depth dimension to facilitate the reduction of facial pose.
- a convolutional layer with reduced number of parameter that exploits facial symmetry and learns from only one half of the face.
- an illumination and pose invariant emotion recognition classifier that produces state-of-the-art classification performance on images taken in both, controlled and unconstrained environments.

The pose invariant GASCA model is trained in different stages using Gradual-GLW. Each shallow autoencoder learns to gradually reduce facial pose, or keep it the same if it is already smaller than the desired target. This process is repeated until reaching a facial pose of 0 degrees. Effectively, the search space for the upper layers is greater than that of the deepest layer, which only has to learn one facial pose of 0 degrees. Similarly to the illumination invariance model from Chapter 4, by only dealing with frontal images, the search space for the fully connected layer in the CNN is dramatically reduced.

The motivation to reduce the pose in facial expression images comes from the observation that in non-frontal faces —i.e. faces with pose greater than 0 degrees —much of the information essential for emotion recognition is nonexistent. Moreover, the more variations in facial pose the larger the data distribution, and the more difficult for a neural network to provide good generalization due to the high dimensional search space. In addition, for real time emotion recognition in unconstrained environments, it is difficult to obtain images without facial pose. Consequently, this chapter explores the development of an emotion recognition model, GASCA_1, capable of dealing with faces with a facial pose of up to 60 degrees. A second model, GASCA_2, which combines the illumination and pose invariant models into one is also presented. The GASCA_1 model is only trained to learn a pose invariant hidden representation in order to demonstrate this novel training approach in more detail, whereas GASCA_2

combines the findings gathered by training GASCA_1 and the illumination invariant SCAE models from Chapter 4.

The deep learning architectures proposed here utilize the findings gathered in the previous chapter, namely the Gradual-GLW training algorithm and the concept of pre-training a deep CNN as a SCAE where the input image and target reconstruction image lie in different distribution spaces. The pose invariant model produces state-of-the-art classification results on multi-pose emotion recognition from facial expressions. The following section of this chapter presents the experimental setup.

5.2 Experimental Setup

5.2.1 Multi-pose Facial Corpus: Multi-Pie

Two pose invariant GASCA models are trained on the MultiPie database of faces described in Chapter 4. The same image pre-processing approach for face detection and dimensionality reduction is followed. The MultiPie corpus contains images with facial pose, φ, at the following angles: $\{0, \pm 15, \pm 30, \pm 45, \pm 60, \pm 75, \pm 90\}$. However, since the faces at $\pm\{70, 90\}$ degrees contain very little facial features useful for emotion recognition, these are not considered in this work.

In the GASCA models each shallow autoencoder deals with a given estimate facial angle at a time, whether facing left or right, i.e. negative or positive angles, and all the facial images with pose equal or smaller than the target angle α —where $\alpha \in \{0, \pm 15, \pm 30, \pm 45\}$ and $n = dim(\alpha)$. Note that images at ± 60 degrees are only used as input and never as target images.

Recall that in the MultiPie corpus all images for a given subject at a given session were taken simultaneously, resulting in multi-pose multi-illumination images of the same subject. All the resulting images with the same pose and varying illumination are stored in the same folder. Therefore, for each session for a given subject there are

9 folders, i.e. one folder for every angle $\{0, \pm 15, \pm 30, \pm 45, \pm 60\}$, each containing 19 images with different illumination.

Accordingly, the dataset is divided into $n-1$, i.e. 4 in this work, subsets containing facial expression images with a given facial angle, whether positive or negative, and all the images with a smaller facial angle. For instance, subset A_1 contains all the images with $\{0, \pm 15\}$ degrees. A_2 contains all images at angles $\{0, \pm 15, \pm 30\}$, thus $A_1 \cap A_2$, and so forth. However, as done for the illumination invariant SCAE model from Chapter 4, the reconstruction target for each shallow autoencoder is not the same as the input image. In the case of the GASCA$_1$ model, the reconstruction target x_μ is either the input image x itself or the image taken simultaneously but from a smaller angle. For instance, in the first shallow autoencoder trained on A_{n-1}, the images of the same subject at -60 and $+60$ degrees are used as input and the images at -45 and $+45$ are used as target. And since all the other images already have a pose closer to 0 degrees they are used as input and target. Note that the increase in angle, for the images with a negative angle, or decrease, for the ones with a positive angle, is done in intervals of 15 degrees due to the structure of the Multi-Pie corpus. Formally this is defined as:

$$x_\mu = \begin{cases} x_{\varphi - d} & , \quad \text{if} \quad 0 < \alpha < \varphi \\ x_{\varphi + d} & , \quad \text{if} \quad \varphi < \alpha < 0 \\ x_\varphi & , \quad \text{if} \quad |\varphi| \leq |\alpha| \end{cases} \tag{5.1}$$

where α denotes the desired target pose and d denotes the change in pose by degrees: 15 degrees in this work. Each subset is further split into 70% training and 30% validation subsets, X and $\tilde{X}$. The creation of these subsets following the methodology described herein plays a vital role in learning illumination and pose invariant hidden representations.

For the GASCA$_2$ model, the subsets are created in the same manner except that instead of simply using the version of an image with a smaller angle as the target, the target is the image with a smaller angle and with relative luminance Y level closest to the mean as done in Section 4.2.4 of Chapter 4. This ensures that the GASCA$_2$ model learns a hidden representation h that is both illumination and pose invariant.

Figure 5.1: Top row: sample faces with +30 degrees pose. Bottom row: faces at +15 degrees used as target for the top row images in the $GASCA_1$ model. Middle image in bottom row has relative luminance closes to the mean and is used as target for all the images in the top row in the $GASCA_2$ pose and illumination invariant model.

Figure 5.1 shows sample images used as input (top row) and images used as desired target reconstructions (bottom row) for the $GASCA_1$ model. In the $GASCA_2$ model, the middle image in the bottom row is used as target for all the top row, as well as for the images in the bottom row when $|\varphi| \leq |\alpha|$. Note that the creation of the training and validation subsets in this manner plays a vital role in learning illumination and pose invariant hidden representations of the input images, and both GASCA models heavily rely on it.

5.2.2 Facial Expression Corpora

The pose invariant GASCA model is used to initialize a classifier model, CNN_1, which is fine-tuned and tested on the KDEF corpus, also introduced in Chapter 4. However, in this case frontal and images at ± 45 degrees are used. No other publicly available datasets with multiple poses have facial expression labels.

The pose and illumination invariant $GASCA_2$ model is used to initialize a second classifier, CNN_2. This model is also fine-tuned and tested on the KDEF corpus. In addition, due to the lack of publicly available data taken in realistic environments with multi-pose and varying illumination, as well as labels for the emotions being expressed, CNN_2 is also tested on a corpus collected using a NAO robot as we pre-

sented in [78] and referred to as NAOFaces hereafter. A total of 196 images from 28 participants were collected in three sessions and two different classrooms using NAO, a 58 centimeters tall humanoid robot with a 1.22 megapixel camera capable of capturing images at 30fps. Participants include 21 males and 7 females between ages 18 and 55, are either students or staff members from at least five different ethnic backgrounds.

During data collection of the NAOFaces corpus, participants were asked to express one of seven emotions at a time as natural as possible and no further instructions were provided. This resulted in participants sitting across from NAO at varying distances, different heights, and looking in different directions, i.e. varying facial pose and tilt. Moreover, no other factors were controlled: participants wore glasses, scarves, and hats in some cases. Lighting was not controlled and the classrooms had windows allowing natural light in, resulting in varying image luminance. The resulting facial expression images were labeled by at least three independent parties and images labeled as the same emotion unanimously, a total of 121 images, were added to the final corpus. For the final corpus, faces were cropped and the resulting images were gray-scaled and normalized to zero mean unit variance as done for all the other corpora used in this research. Note that none of these images are used for fine-tuning.

5.3 ConvMLP and HalfConv layers

One of the main advantages offered by CNNs over MLPs is their ability to self-learn a translation invariant downsampled feature vector that highlights salient features and retains spatial information through filter kernels. This is particularly beneficial for visual processing tasks where spatial information plays a crucial role in identifying features of interest, e.g. the shape of the mouth and eyebrows for emotion recognition.

Equally important, CNNs are significantly less computationally expensive than MLPs, due to the exponentially smaller number of parameters. Although these ad-

vantages offered by CNNs often yield high accuracy rates, CNNs are constrained to preserve the spatial structure of images and therefore are not suitable to reduce or increase facial pose: since every output value produced by convolutional layers is the results of the dot product between a filter kernel and a small view of the input image, the pixel values can only be shifted within the space covered by the filter kernel. Normally, filter kernels tend to be small in order to capture small salient features. In this research, 3×3 kernels have demonstrated to be the most efficient. Due to the small are covered by these filters, a pixel value can only be shifted two spaces in a given direction, which is not enough to shift facial features to a frontal view. For instance, in a 100×100 facial expression image with an estimated pose at -60 degrees, facial features like the nose and left eye lie in the region covered by pixels 1 to 25 and need to be shifted between 10 to 25 places over the x axis in order to obtain a frontal view.

Using larger kernels that can capture a larger area and allow spatial information to be shifted 10 to 25 places over, results in a loss of smaller salient features and a decrease in classification accuracy for emotion recognition. Moreover, because in many cases some facial features are not visible if the pose is greater than ± 46 degrees, a convolutional layer will struggle to fill in the missing information considering that its primary goal is to highlight salient features and retain spatial information. One alternative is to substitute convolutional layers with fully connected layers, i.e. use an MLP instead of a CNN for every shallow autoencoder. However, MLPs do not take into consideration spatial information, are more prone to overfitting, and are more difficult to train due to the exponentially higher number of parameters. For example, consider the SCAE model from Chapter 3, which has been used as the base model for all the other architectures in this thesis. The SCAE model utilizes 20 filter kernels in the first convolutional layer, which takes $1 \times 100 \times 100$ gray scaled images. The weight matrix for this layer is of size $20 \times 1 \times 100 \times 100$. If this is to be replaced with a fully connected layer that can keep the input image at the same size —the image needs to be kept at the same size to be able to measure the loss between the input and target images—it would require $10,000$ hidden units and a weight matrix of size $10,000 \times 10,000$, over 500 orders of magnitude larger than the CNN layer.

To overcome the limitations imposed by convolutional kernels and fully connected layers, and at the same time exploit the advantages offered by both, this section of the thesis introduces a hybrid layer that combines both approaches. The most straight forward to accomplish this is by simply placing an MLP after the convolutional layer. And, by having a smaller number of hidden units in the MLP than the number of features produced by the convolutional kernels, there would be no need for down-sampling layers such as average or max pooling or convolutional layers with a stride greater than one, which often result in the loss of important information. However, because convolutional layers normally employ a high number of convolutional kernels in order to extract several salient features, thus adding an extra dimension in the weight matrix, this approach would require a significantly large matrix weight W. Accordingly, W would need to have a connection weight for each feature in the feature maps produced by convolutional kernels, resulting in a large number of learnable parameters, increased computational cost, and increased training difficulty.

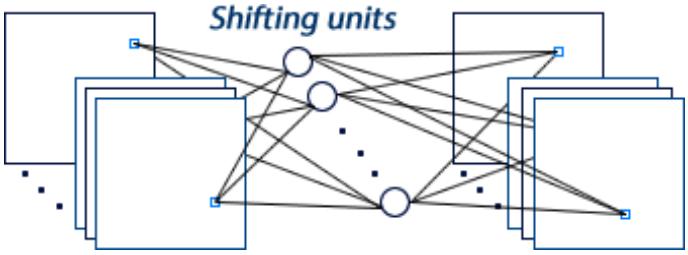

Figure 5.2: ConvMLP layers illustration. Connection weights for the *shifting units* are shared between all the feature maps.

In contrast, the novel layer presented here, referred to as ConvMLP hereafter, shapes the resulting feature map produced by a convolution operation with a fully connected layer that is shared between all the resulting feature maps. Refer to Figure 5.2 for a pictorial description. Given an input image I and a filter kernel K with $m \times n$ dimensions, and a second weight matrix W, the output of ConvMLP layers is defined as:

$$C(i,j) = W\big((I * K)(i,j)\big) \tag{5.2}$$

where:

$$(I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n) \tag{5.3}$$

Just as in empirical convolutional layers, the non-linearity is provided by a ReLU activation function, extending the above Equation to:

$$y = \max(0, C(i, j)) \tag{5.4}$$

In this formulation of ConvMLP layers, during the forward pass, the weight matrix W is used to shape every feature map produced by the convolution operation and is updated only once using back propagation. Sharing this layer across the third dimension—not taking into account the batch dimension for simplicity—the size of W in the scenario described above is only 100×100 as opposed to $200,000$ without weight sharing, resulting in a dramatically smaller number of parameters. This also ensures that the $shifting$ layer learns to shift all the features highlighted in every feature plane in the same manner. Notice in Figure 5.2 how the pixels on the second feature map are at a different location.

In addition to ConvMLP layers, and in order to support the pose invariant training approach and models presented in this chapter, a second convolutional layer is introduced here. This novel layer, referred to as HalfConv hereafter, exploits facial symmetry present in face images with an estimated pose of zero degrees. HalfConv layers slice the input vector vertically in half. The half containing all the facial features belonging to the left side of a face is then used as input for a convolutional layer that has half the number of parameters than an empirical convolutional layer. The resulting feature map is then simply mirrored across the y axis. Just like empirical convolutional layers, HalfConv layers can compute multiple feature planes. Figure 5.3 illustrates the concept of HalfConv layers. Notice that the layer takes half of an image as input and produces feature planes with the full image.

Figure 5.3: HalfConv layers illustration.

When applied to face or facial expression images, HalfConv layers give up some important information on the right edge of the input image, which in effect corresponds to the features in the middle of a face. This is due to the nature of the convolution operation, which convolves a kernel across an input image, resulting in a feature plane with smaller dimensions than the input image. For this reason, HalfConv layers enforce zero padding p on right side edge of the input image to allow the filter kernel to capture the features closer to the edge. Their output is then defined by:

$$C(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m,(j+p)-n) \tag{5.5}$$

where $p = \frac{j}{2} + 1$. Then every resulting feature plane is reflected over the y axis, resulting in a full image. Note that padding p is enforced to avoid losing features at the edges of the image.

The main advantage offered by HalfConv layers is the reduced number of learnable parameters, which in effect results in easier and faster training. Because the only extra operation required by this layer is simply mirroring a feature vector vertically, HalfConv layers are significantly less computationally expensive than empirical convolutional layers. Furthermore, because this layer only deals with frontal faces, there is no need to employ any shifting neurons. Note that these layers are only suitable for cases where symmetry is existent in the input image or is desired in the resulting feature plane. Therefore, in the GASCA model, these layers are only used when $\alpha = 0$.

5.4 Generative Adversarial Stacked Autoencoders

In previous chapters of this thesis, it was established that the input and target reconstruction images used to train an autoencoder do not need to be same. This approach allows a neural network to learn a mapping from an input image to a target image that may or may not lie in a different distribution. In effect, the network learns to impose a distribution on the input data to produce reconstructions that resemble the

desired target. Adversarial autoencoders can facilitate this task as they uniformly impose a data distribution on the code vector, i.e. the hidden representation produced by the encoder element, to generate realistic reconstructions. Moreover, adversarial autoencoders are designed to produce very realistic reconstructions with minimal loss of information. Accordingly, this chapter builds on this framework, along with the findings from earlier chapters, and introduces a novel generative adversarial stacked convolutional autoencoder (GASCA) model. This framework is employed in order to gradually reduce facial pose to zero degrees while at the same time retaining all the salient features that are essential for emotion recognition.

Recall from Chapter 2, GANs are composed of two networks: a generative model G and a discriminator model D [28]. Both models are trained by playing a min-max adversarial game where the discriminator model tries to determine if a given sample is from the generator or the dataset. In contrast, the generator maps samples z from a prior distribution $p(z)$ and maps it to the data space. Adversarial autoencoders follow a similar approach where the generator is an autoencoder that maps an input x to a latent representation z that lies in an aggregate posterior distribution $q(z)$ and back to a reconstruction y which is an approximation of x. The discriminator network in this framework attempts to determine if a sample has been drawn from a prior distribution $p(z)$ or from the latent distribution $q(z)$.

In the GASCA model, the discriminator attempts to tell whether a sample comes from the training dataset or if it is a reconstruction produced by the autoencoder. Let x_φ be a sample from the data distribution $p_d(x_\varphi)$ and x_μ the sample from the data distribution $p_d(x_\mu)$ used as the desired target reconstruction defined according to Equation 5.1. The autoencoder G model learns to map x_φ to a latent space z, note that this is not an aggregate posterior as in conventional adversarial autoencoders, and back to a reconstruction y that resembles x_μ and lies in the distribution $q(y)$. The discriminator D attempts to differentiate between y and x_μ.

The conventional adversarial autoencoder framework [27] imposes $p(z)$ —often a Gaussian distribution—on $q(z)$ by estimating the divergence between q and p. This

imposition can be used to produce reconstructions with specific features. However, in this work, the objective is to produce reconstructions that are as close as possible to the desired target image x_μ. Consequently, instead of imposing random noise on the hidden representation vector, the GASCA model imposes $p_d(x_\mu)$ on $q(y)$ in the following way:

$$q(y) = \mathbb{R}_{x_\mu} q(y|x_\mu) p_d(x_\varphi) dx_\mu \tag{5.6}$$

This formulation assists in the reduction of facial pose. Furthermore, the GASCA model is trained in a greedy layer-wise manner using the Gradual-GLW training method proposed in Chapter 4. By employing Gradual-GLW, the GASCA model is able to overcome the added difficulty of training GANs as it is often the case.

With this formulation, the discriminator model D is optimized to rate samples from $p_d(x_\mu)$ with a higher probability, and samples from $q(y)$ with a low probability. Formally this is defined as:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \Big[\log D(x_\mu^{(i)}) + \log \big(1 - D(G(y^{(i)}))\big) \Big] \tag{5.7}$$

where $x_m u$ is defined according to Equation 5.1. Note that since the objective is to generate an image with a smaller facial pose, D never sees the input image x_φ.

The objective of the autoencoder model G, which in term plays the role as the generator, is to convince the discriminator model D that a sample reconstruction y was drawn from the data distribution $p_d(x_\mu)$ and not from $q(y)$. This optimization is done according to:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log \Big(1 - D\big(G(y^{(i)})\big)\Big) \tag{5.8}$$

5.5 Unsupervised Feature Learning

Two pose invariant models are proposed in this chapter. Both models are constrained by the theoretical convolution and adversarial learning methods described earlier in

Figure 5.4: Visualization of the first shallow autoencoder in the GASCA model.

this chapter, and rely on the creation of a training set as described in the experimental setup section of this chapter. The first model, GASCA, is designed to address pose invariance, whereas the second model, GASCA_2 incorporates the illumination invariance findings gathered in Chapter 4 with the pose invariant methodology of this chapter. Figure 5.4 illustrates how the first layer in the GASCA models is trained. For a full description of the topology for both networks in the GASCA model refer to Table A.1 in Appendix A.

Both models utilize the same topology as the SCAE model in Table 3.2 from Chapter 3. However, the first three convolutional layers are replaced with ConvMLP layers and the last one is replaced with a HalfConv layer. Moreover, both models are trained in a greedy layer-wise unsupervised fashion using Gradual-GLW. The Gradual-GLW method from Chapter 4 is modified to comply with the adversarial learning paradigm as showing in Table 5.1.

Intuitively, it makes sense to encode x_μ using D to learn D^k. However, because the features learned by D and G are similar, and since D does not use a fully connected layer like empirical CNNs, it is unnecessary to add an extra step. Fine-tuning D for classification leads to D being good at differentiating between samples drawn from $q(y)$ and those from $p_d(x_\mu)$. Therefore, forcing D to improve its ability to generate more realistic images in the next step, i.e. when the next shallow autoencoder is trained and the stack is fine-tuned again.

Every shallow autoencoder in the GASCA models is trained for 100 and fine-tuned for 20 epochs. G is optimized using ADAM whereas D employs SGD with Nesterov

Table 5.1: Gradual-GLW Semi-supervised Adversarial Training.

Given a training set X and validation set $\tilde{X}$ each containing input images x_φ and target images x_μ, m shallow autoencoders, an unsupervised feature learning algorithm $\mathcal{L}$ —see Table 5.2 —which returns a trained shallow autoencoder and a discriminator model, and a fine-tuning algorithm $\mathcal{T}$ —see Table 4.3 in Chapter 4: train D^1 and G^1 jointly with raw data and add them to their corresponding stacks G and D. For the remaining autoencoders and generator models: encode X and $\tilde{X}$ using the encoder layers ξ from the stack G. Create a new discriminator D^k and train together with the new autoencoder G^k and add them to their corresponding stacks. Fine-tune G on raw pixel data. Forward propagate $x_\varphi \subset X$ through G and use the resulting features, along with x_μ, to fine-tune D for binary classification. Note that D does not have a fully connected as empirical CNNs.

$[G^1, D^1] \leftarrow \mathcal{L}(G^1, D^1, X, \tilde{X})$
$G \leftarrow G \circ G^1$
$D \leftarrow D \circ D^1$
for $k \leftarrow 2, \ldots, m$ **do**
  $[\xi, \delta] \leftarrow D$
  $[X_g, \tilde{X}_g] \leftarrow \xi(X, \tilde{X})$
  $[G^k, D^k] \leftarrow \mathcal{L}(G^k, D^k, X_d, \tilde{X}_d)$
  $G \leftarrow G^{(k)} \circ G$
  $D \leftarrow D^{(k)} \circ D$
  $G \leftarrow \mathcal{T}(G, X, \tilde{X})$
  $X_\varphi \leftarrow G(x_\varphi)$
  $D \leftarrow \mathcal{T}(D, \{X_\varphi, x_\mu \subset X\})$
end for
Return G, D

Table 5.2: Gradual-GLW adversarial procedure from Table 5.1

Given a training dataset X with m mini-batches of size b, an autoencoder model G and discriminator model D both with weight matrices W_g and W_d, an absolute value cost function $loss$: train G and D jointly such that:

$V(W_d) \leftarrow \frac{2}{Nin+Nout}$
$V(W_g) \leftarrow \frac{2}{Nin+Nout}$
for $k = 1, \ldots, M$ **do**
  for $n = 1, \ldots, m$ **do**
    $[x_\varphi, x_\mu]_n \subset 1, \ldots, m \leftarrow random(X, b)$
    $y_g \leftarrow predict(x_\varphi, G)$
    $L_g \leftarrow loss(x_\mu, y_g)$
    $G \leftarrow update(G, L_g)$
    $p_\mu \leftarrow predict(x_\mu, D)$
    $L_{p_d(x_\mu)} \leftarrow loss(1, p_\mu)$
    $D \leftarrow update(D, L_{p_d(x_\mu)})$
    $p_y \leftarrow predict(y_g, D)$
    $L_{q(y)} \leftarrow loss(0, p_y)$
    $L_{adversary} = L_{p_d(x_\mu)} + L_{q(y)}$
    $L_{minimax} \leftarrow loss(1, p_y)$
    $L = L_{minimax} + L_g$
    $MM_{L_g} \leftarrow lossGrad(1, p_y)$
    $MM_g \leftarrow Grad(y_g, MM_{L-g}, D)$
    $G \leftarrow update(G, MM_g)$
    $Adam(L, G)$
    $SGD(L_{adversary}, D)$
  end for
end for
Return G, D

momentum. The initial learning rates for each individual shallow autoencoder in G were set to $\lambda \in \{0.1, 0.3, 0.5, 0.7, 0.75\}$ and annealed by a factor of 0.01 using Equation 3.16. Since D learns faster than G, the shallow autoencoders employ smaller learning rates: $\lambda \in \{0.01, 0.03, 0.5, 0.07\}$ and are not annealed. During fine-tuning the stacks G and D use a learning rate of 0.01 and are annealed using a factor of 0.3.

5.6 Emotion Recognition

As done for the illumination invariant SCAE models in Chapter 4, once a GASCA model is trained and fine-tuned for reconstruction, which is a regression problem, both the discriminator model D along with the decoder element g_D of the generator D model are discarded. The resulting encoder model, which produces a pose invariant—and illumination invariant, in the case of the GASCA_2 model—feature vector z, is then used to initialize a convolutional classifier.

The GASCA model is used to initialize CNN_1 and is fine-tuned on the training subset of the KDEF corpus. This classifier is fine-tune for 10 epochs. The GASCA_2 model is used to initialize two classifiers, namely CNN_2a and CNN_2b. The former is also fine-tuned on the training subset of the KDEF corpus, whereas the latter is fine-tuned on the CFE corpus which is composed of the CK+, JAFFE, KDEF, and FEEDTUM corpora. Note that the CFE corpus contains the images with multiple poses from the KDEF corpus and since the illumination invariant model from Chapter 4 achieved state-of-the-art classification performance on this corpus we use the entire corpus for fine-tuning in an attempt to improve the models generalization performance using more data. Consequently, this model is evaluated on completely novel data: the entire NAOFaces corpus.

As opposed to the models in previous chapters, which employ a fully connected layer after the last convolutional layer, the classifiers in this chapter map the resulting feature planes produced by the last convolutional layer, which is a HalfConv layer, directly to an output SoftMax layer for classification, as done in [14].

The CNN_2a model is fine-tuned for 10 epochs and because the CFE corpus has more images CNN_2b is only fine-tuned for two epochs. Since the stacked autoencoders are optimized using ADAM, all classifiers are fine-tuned also using ADAM and a learning rate of 0.1. Using a different optimizer like SGD for fine-tuning would lead to the gradients changing dramatically and require a longer fine-tuning process.

5.7 Pose Invariant Reconstruction Results

The novel pose invariant Generative Adversarial Stacked Convolutional Autoencoder models proposed in this chapter are trained to gradually reduce facial pose. A shallow autoencoder is created to deal with specific pose interval, 15 degrees, in any given direction. Each shallow autoencoder is trained using adversarial learning, where the autoencoder is the generator model G^k and a new shallow CNN is the discriminator D^k. Once both models are trained jointly the resulting models are added to their corresponding stacks G and D and fine-tuned further. In the case of G it is fine-tuned on raw pixel data where the input is x_φ and the target reconstruction is x_μ.

In the case of D, x_φ is passed through G and the resulting reconstruction y is assigned the label 0. x_μ is assigned the label 1 and D is fine-tuned for classification. Initially, since the reconstructions produces by G are significantly different than the target reconstruction images, D learns to classify these two relatively fast. However, as G becomes better at producing reconstructions, the classification performance of D drops significantly. Once this happens the training of both models is halted since it means that it is difficult to differentiate between y and x_μ and D ends up making random decisions.

As it can be observed in Figure 5.5, the pose invariant GASCA model manages to reduce facial pose in facial images with an estimated pose of up to ± 60 degrees. It can also be observed that on the images with pose of ± 60 degrees half of the face is not visible, yet the pose invariant model manages to fill in the missing information, and more importantly keeps the shape of facial shapes which are important for emotion

Figure 5.5: Top row: input images x_φ to the GASCA model with estimated facial poses at $+60, +45, +30, +15, 0$ degrees. Bottom row: corresponding reconstructions y produced by the GASCA model with an estimated pose at ~ 0 degrees.

recognition: eyes, eyebrows, mouth, nose, cheeks, among others. Nonetheless, the greater the pose in x_φ the poorer the quality of the reconstruction y. This is justified by (i) the fact that the GASCA model has to compensate for missing information, (ii) the fact that only one layer is trained specifically to deal with that particular facial pose, (iii) the smaller the pose the more the images get seen by every layer in G during training, and (iv) increased network depth.

If the shallow autoencoder at step $k = 1$ fails to learn a pose invariant feature vector, the shallow autoencoder at step $k = 2$ will struggle even more to learn a pose invariant feature vector, and so forth. Gradual-GLW training proposed in Chapter 4 greatly helps to address this issue by allowing inter-layer fine-tuning, which helps strengthen the weight connections between D^k and D^{k+1}. One alternative to Gradual-GLW training is GLW. However, as seen in Chapter 4 it is prone to high error accumulation and poor image reconstructions. Similarly, G could be trained as a single unit, i.e. training all the layers at once. Training in this manner is a naive approach considering that G has a large number of parameters and finding the right initialization parameters is a challenge in itself. Moreover, as seen in Chapter 3, joint training usually requires an exponentially higher number of epochs. These observations highlight the vital role played by adversarial learning to obtain pose invariant feature vectors.

One of the main remarks observed in the reconstructions is that although these

retain all the important salient features, they are visually different than the the input images. These reconstructions could be improved by unsupervised fine-tuning of G for a significantly longer number of epochs. Likewise, secondary methods such as super resolution CNNs [79] could be used to improve the visual quality of the reconstructed images. However, because the objective of this research is to only learn a pose invariant feature vector z that can be used for emotion recognition, the quality or resolution of the reconstructions is trivial.

The ConvMLP layers proposed in this chapter are fundamental for the reduction of facial pose. An empirical convolutional layer is unable to shift facial features due to the restrictions imposed by the size of filter kernels. Every feature in a feature plane produced by a convolutional layer is produced taking into account only a small area in the input image. Therefore, they are unable to shift facial features a given number of places within the image space. Moreover, MLPs, which are composed of fully connected layers, have a significantly larger number of parameters and are prone to overfitting. Likewise, since a convolutional layer with filter kernels with height and width greater than one, followed by a 1×1 convolutional layer, is mathematically equivalent to a fully connected layer [80], these could theoretically be used to reduce facial pose. Yet, when evaluated individually these are still constrained by the spatial structure of an image.

One of the main advantages offered by ConvMLP layers is that the number of $shifting$ neurons can be adjusted as needed. In the GASCA models, every ConvMLP layer only employs 100, which are enough to reposition facial features and eventually reduce facial pose. Another advantage offered by ConvMLP layers is that they can be used for dimensionality reduction by mapping a feature plane to a smaller feature plane. Although, this is not evaluated in this research.

As illustrated in Figure 5.5, the reconstructed images also do not have a horizontal line diving the face in two, as it would be expected due to the use of HalfConv layers. When visualizing the feature planes produced by these layers, the line is somewhat visible. However, because in the final stack G this layer is followed by all the layers

in the decoder stack of G, and since the line is not visible in the target reconstruction images, it vanishes during fine-tuning.

5.8 Pose Invariant Emotion Recognition Results

One of the main advantages of pre-training the CNN as a GASCA mode, is that the data distribution $p_d(x_\mu)$ is mapped to a smaller distribution $q(z)$, where the feature vector z is pose invariant. Accordingly, the search space for the classification layer in the CNN model is significantly smaller since it only deals with one facial pose. This also leads to faster fine-tuning of the CNN. The GASCA model is used to initialize CNN_1, which is fine-tuned on the KDEF training subset, which contains multi-pose facial expression images.

Table 5.3: Classification performance (96.810%) of the CNN_1 model on the KDEF corpus.

	A	D	F	H	N	Sa	Su
A	94.44	1.59	1.59	0.00	0.79	1.59	0.00
D	0.00	97.60	0.00	0.00	0.00	2.40	0.00
F	000	0.79	89.68	0.79	0.00	3.97	4.76
H	0.00	0.00	0.00	100.00	0.00	0.00	0.00
N	0.00	0.00	0.00	0.00	100.00	0.00	0.00
Sa	0.79	0.79	0.00	0.00	0.00	98.41	0.00
Su	0.00	0.00	2.42	0.00	0.00	0.00	97.58

As it can be observed in Table 5.3, the CNN_1 model obtains a classification performance of 96.81%. For comparison purpose, the GASCA model introduced in Chapter 3 achieved an accuracy rate of 92.52%, over 4% lower even though it was evaluated only on the images with 0 degrees pose. The difference in performance supports the pose invariant feature learning method presented in this chapter. Nonetheless, the performance on individual classes is consistent for both models.

The GASCA_2 model is used to initialize a second classifier, namely CNN_{2a}, which is fine-tuned on the KDEF. Furthermore, CNN_{2b} is evaluated on the NAOFaces cor-

pus. Note that none of the images in NAOFaces are used of fine-tuning. The $GASCA_2$ model combines the findings obtained in Chapter 4 on illumination invariance unsupervised feature learning and combines them with the findings observed when training the GASCA model. The results are reported in Table 5.4.

Table 5.4: Classification performance (98.070%) of the CNN_{2a} model on the KDEF corpus.

	A	D	F	H	N	Sa	Su
A	96.83	0.79	1.59	0	0.00	0.79	0.00
D	0.00	97.60	0.00	0.00	0.00	2.40	0.00
F	000	0.79	93.65	0.79	0.00	2.38	2.38
H	0.00	0.00	0.00	100.00	0.00	0.00	0.00
N	0.00	0.00	0.00	0.00	100.00	0.00	0.00
Sa	0.79	0.79	0.00	0.00	0.00	98.41	0.00
Su	0.00	0.00	0.00	0.00	0.00	0.00	100.00

As illustrated in Table 5.4, the CNN_{2a} model outperforms the CNN_1 model and obtains a state-of-the-art classification rate of 98.07% on the KDEF corpus. The main differences in performance are observed for classes: surprise, Fear, and Angry, whereas both CNN_1 and CNN_{2a} obtained the same classification accuracy for the remaining classes. Because both models are trained using a relatively similar approach, it is hypothesized that these discrepancies in classification performance are due to these three classes containing more images with varying image luminance, thus the pose and illumination invariant model is able to generalize better.

One important observation is that, when looking at the missclassified images for a given class, on average 40% of them are frontal images, i.e. images with zero degrees pose, and the remaining 60% are those with a pose. However, because the ratio of images with a facial pose is 2 : 1 compared to those without one. This means that on average, more images without facial pose are missclassified. These results and observations are of great importance given that they support the pose invariant pretraining approach presented in this chapter.

Table 5.5: Classification performance (81.36%) of the CNN_{2b} model on the NAOFaces corpus.

	A	D	F	H	N	Sa	Su
A	92.86	7.14	0.00	0.00	0.00	0.00	0.00
D	8.33	75.00	8.33	0.00	0.00	8.33	0.00
F	9.09	0.00	81.81	0.00	0.00	0.00	9.09
H	0.00	0.00	0.00	100.00	0.00	0.00	0.00
N	3.85	0.00	3.85	15.38	57.69	11.54	7.69
Sa	9.09	0.00	18.18	0.00	0.00	72.72	0.00
Su	0.00	0.00	10.53	0.00	0.00	0.00	89.47

The CNN_{2b} model is fine-tuned on the entire CFE corpus and evaluated on the entire NAOFaces corpus. This classifier achieves an accuracy rate of 81.36%. As we reported in [78], when using the illumination invariant training approach from Chapter 4, the performance achieved is 73.55%, that is 7.81% lower than the pose and illumination invariant model presented here. The superiority of the pose and illumination invariant model can be justified by the varying poses and tilt of faces in the NAOFaces corpus, which was collected in unconstrained environments.

As shown in Table 5.5, not a single image from the other classes was confused with Neutral. This particular score is significant taking into account that all emotions derive from a neutral state, often resulting in low precision scores.

Despite the increase in performance offered by the CNN_{2a} on the NAOFaces corpus, the classification performance offered by this model is not ideal. This is attributed to one major factor: cultural differences. Because the model was trained solely on images from Caucasian people, the model has never learned to adjust to cultural difference. The NAOFaces corpus contains images of people from at least five different backgrounds including: Asian, Arab, Black, Irish and Hispanic, among others unrevealed ones. In effect, because people from different ethnic backgrounds express emotions differently [81], the classifier should be trained with images of participants from a wide range of ethnic backgrounds and cultures. Moreover, despite expressing emotions differently, people from different backgrounds may have different facial features, resulting in different spatial information, which is what the CNN

models take into account when learning to extract salient features. Nevertheless, the results obtained by the CNN_{2b} are remarkable for recognition in unconstrained environments.

5.9 Comparison Against State-Of-The-Art

Table 5.6: Classification performance comparison on the KDEF corpus: ResNet-34 —state-of-the-art classifier; CNN_1 —pose invariant classifier proposed; CNN_2 pose and illumination invariant classifier proposed.

	$Resnet34$	CNN_1	CNN_{2a}
A	84.127%	94.444%	96.825%
D	85.600%	97.600%	97.600%
F	73.810%	89.683%	93.651%
H	98.413%	100.000%	100.000%
N	90.400%	100.000%	100.000%
Sa	84.921%	98.413%	98.413%
Su	95.161%	97.581%	100.000%
$Total$	87.472%	96.810%	98.070%

Due to the lack of contemporary work designed explicitly for pose invariant emotion recognition, the methods proposed in this work are compared against one of the most common and state-of-the-art classifiers: a ResNet [14]. ResNet models use an identity shortcut —a skip connection that skips one or two layers and allows a given layer to receive as input the output of the previous layer along with the output of the second or third layer before —that facilitates the flow of information, enabling large network depth. Accordingly, a ResNet-34, i.e. with 34 parametrised layers, is trained using SGD, a momentum of 0.9 and learning rate of 0.1. This model is trained for 100 on the training subset of the KDEF corpus and achieves an accuracy rate of 87.472% on the test subset, as illustrated in Table 5.6. Note that even though the SCAE model introduced in Chapter 3 achieved 92.52% on the KDEF corpus, those results are only reported on frontal faces without facial pose. On the contrary, all the models in this chapter are evaluated on images with multiple poses, hence the marginally lower performance of the ResNet model.

As seen in Table 5.6, the pose and illumination invariance model, CNN_{2a} outperforms the state-of-the-art classifier ResNet-34 model by over 10%. Similarly, it outperforms CNN_1 marginally, supporting the pose and illumination invariant training approach. The pose invariant GASCA models also have an exponentially smaller number of parameters compared to the ResNet-34 model.

The novelty of this work also arises from combining greedy layer-wise training with adversarial learning. Generative Adversarial Autoencoders are trained jointly as opposed to layer-wise. They impose a random distribution $p(z)$ on the distribution $q(z)$ produced by the encoder element of G, and use the resulting aggregate posterior distribution is mapped to reconstruction y. The discriminator D tries to guess if the sample was drawn from $q(z)$ or $p(z)$. The GASCA models do not use a random distribution and instead use the reconstruction y produced by forward propagating x_φ through G, along with the target image x_μ as input for the discriminator. The generator G is optimized to reduce the distance between y and x_μ. By fine-tuning the stacks G and D at every step k, both models become better at their respective job. By improving the ability of D to differentiate between y and x_μ, G is forced to produce remarkable reconstructions and learn an encoder function that produces downsampled pose invariant feature vectors.

In terms of work on pose reductions, a similar model was proposed by [61]. However, the authors focused on face detection and their model does not make use of Convolutional Autoencoders and instead uses MLPs, which are prone to overfitting when applied to this problem. Furthermore, because their model does not take into account spatial information, it is unable to retain salient features that are essential for emotion recognition. Whereas the GASCA models are able to retain facial features, or compensate for missing information when this is not present in the image. Additionally, the GASCA_2 model also takes into account illumination and produces an illumination and pose invariant feature vector.

One of the main constrains of the pose invariant unsupervised training method proposed in this chapter, is that it relies on the availability of multi-pose facial expres-

sion images. Although this is a common limitation of deep learning models, which require large amounts of labeled data to learn meaningful representations.

By demonstrating that in an autoencoder model it is possible to map an input x to a hidden representation z and back to a reconstruction y that resembles a desired target x_μ and $\neg\Box(x = x_\mu)$, it can be established that this training approach can, theoretically, be used to learn a mapping from and input to a desired target that lies in a completely different distribution. For instance, this method could used to learn to reduce face or object rotation.

5.10 Chapter Conclusion

This chapter of the thesis has introduced a novel pose invariant facial expression recognition model. A CNN classifier is pretrained as a Generative Adversarial Stacked Convolutional Autoencoder in a greedy layer-wise semi-supervised fashion. The GASCA model learns to map an input image containing a face, with an estimate pose φ, to a hidden representation z with an estimated pose of 0 degrees. Once the GASCA model is trained, the encoder elements is used to initialize a CNN model which is fine-tuned for classification.

The outstanding performance of the GASCA models relies on four concepts: (i) the Gradual-GLW method from Chapter 4 combined with Adversarial Learning, (ii) the ConvMLP layers with $shifting$ neurons, and (iii) the HalfConv layers which take exploit of facial symmetry, and (iv) multi-pose facial expressions data. Combined with Gradual-GLW and Adversarial Learning, the pose invariant methodology presented in this chapter produces state-of-the-art classification performance on multi-pose facial expression corpora. Moreover, the GASCA model produces reconstruction with very small errors and is able to generalize on unseen data. However, it is difficult to compare against other methods since there is limited literature on pose invariant emotion recognition.

The success of the pose invariant models is in part due to ConvMLP layers, which learn salient features and shift them as needed to reduce facial pose. HalfConv layers also play an important role as they reduce the number of learning parameters. HalfConv layers were inspired by the CEN model presented in Chapter 3, which splits the input images in half to simplify feature learning. The main limitation of HalfConv layers are bounded by the assumption that the input is symmetrical across the y axis, which may not always be the case.

To the best of the author's knowledge, this is the first approach that combines a greedy layer-wise training method with adversarial learning. Moreover, this is also the first approach to solely focus on pose invariant emotion recognition. Accordingly, this work is a step forward for the domain of emotion recognition in unconstrained environments. Nonetheless, this and all the emotion recognition models presented in this thesis, rely on one important pre-processing step: face detection. Accordingly, the following chapter looks at the implementation of a face detection model that overcomes some of the limitations of contemporary face detectors, such as their inability to deal with nonuniform data.

Chapter 6

Deep and Reinforcement Learning for Face Detection

6.1 Introduction

The previous chapters of this thesis focus on the development of deep learning models for emotion recognition from facial expressions. Inherently, these models rely on various image pre-processing methods, such as face detection and dimensionality reduction. Face detection, in particular, eliminates unnecessary information such as background noise, resulting in faster training and better generalization of deep networks.

One of the main disadvantages of relying on face detection algorithms, aside from having to do this step for every new image to be evaluated, is that contemporary face detection methods are not very accurate [82], are highly computationally expensive [63], and often fail to detect faces on images with nonuniform conditions. For instance, as later seen in the results section of this chapter, empirical face detection methods such as the Viola-Jones are prone to changes in illumination and pose, which are the two core concepts addressed in Chapters 4 and 5. Accordingly, recognizing emotions can become unattainable if the face detector fails to detect a face in the first instance. This is a major concern for emotion recognition systems designed to work in real time. As a result, this chapter of the thesis explores the use of deep learning in conjunction

with deep reinforcement learning for face detection.

The novel face detection algorithm proposed here, referred to as DeepFace hereafter, makes use of the illumination invariant SCAE model for image pre-processing, and employs an agent capable of learning through experience and interaction. DeepFace is shown to work on images with very low or very high luminance levels, and on faces with some degree of rotation.

6.2 Motivation

In a real-life scenario, facial expression images will likely contain some degree of facial pose, tilt, or rotation. Although not applied to emotion recognition, facial pose in face recognition is widely studied in the literature [29], [83]. Face tilt is often solved when facial pose is fixed as seen in Chapter 5. However, face rotation is often overlooked. This may be due to the fact that when faces have some arbitrary degree of rotation, the facial features usually remain visible.

Accordingly, DeepFace is designed to deal with face alignment with regard to face rotation. Faces with some degree of rotation of up to ± 45 degrees of rotation are considered in this work as these are the commonly encountered scenarios in real. Moreover, dealing with small rotations helps to illustrate the concept of DRL in face rotation in more details.

The work done by [50], [51], [52] and [53] has demonstrated that Deep-Q learning can be employed for face localization purposes without exhaustively searching the entire image space. Because non-exhaustive search is important for applications designed for use in real time, DeepFace employs Deep-Q learning as the underlying learning paradigm and builds upon contemporary work. Moreover, DeepFace is translation and scale invariant.

Another objective of DeepFace is to overcome some of the limitations of empirical

face detectors, which are often unable to deal with changes in illumination. When learning a pose invariant model in Chapter 5, it was observed that the face detection model from [82] and [84] often failed on images with a facial pose or with low luminance levels. In cases where other models worked [63] on such images, the localization time was very expensive and always surpassed 50 seconds on a 32 core system. Although the face detection model proposed here does not deal with facial pose, the concept of fixing facial pose has already been proved in Chapter 5. Once a face is located, a GASCA model can be used to correct the pose.

6.3 Experimental Setup

DeepFace is trained on the BioID Face Database [85]. This corpus consists of 1521 gray-scaled images of 23 participants illustrating a frontal face view and each image has a resolution of 384×286. Each individual image has a corresponding text file containing manually set coordinates of the participants eyes. In addition, a corresponding text file describing twenty additional points such as the coordinates for the mouth, chin, nose, temples, among others, is also provided. This corpus is divided into 70% training and 30% testing subsets.

In addition to the BioID corpus, a small subset of the testing set of the Multi-Pie corpus is also used for evaluation of the face detection model. The subset includes 100 random images with varying illumination and frontal facial pose. Note that only a small subset is used in order to provided a detailed analysis on the performance of DeepFace on unseen data. This subset is only used for testing. All images in this corpus are gray-scaled and resized to 384×286 using bicubic interpolation as defined by Equation 6.3. Note that these 100 images are excluded from the Multi-Pie corpus used below.

The agent employs an illumination invariant SCAE model like the one presented in Chapter 4 for feature extraction. However, since the BioID corpus does not contain multi-illumination images, a new corpus is created to train the SCAE model. This

new corpus, referred to as MultiFaces hereafter, consists of the Multi-Pie and Yale corpora, as well as as γ corrected versions of the following corpora: CK+, KDEF, FEEDTUM, JAFFE, and the training subset of the BioID corpus. Note that because the Yale and MultiPie corpora already have multi-illumination images. Moreover, when referring to the BioID corpus on its own, it is the version described above used to train the Q-network, for which no γ correction is applied.

By creating such a large corpus with varying illumination and facial expressions, it is ensured that the SCAE model will be able to generalize better on unseen data with varying degrees of illumination. The MultiFaces corpus is also split into 70% and 30% training and validation subsets. The validation subset is used to determine the stopping criteria is illustrated in Table 4.2. Note that the MultiFaces is only used to train the SCAE model and not to evaluate DeepFace. As such, there is no testing subset. All images in this corpus are also gray-scaled and scaled to 384×286.

Every image in all corpora, including testing images, are zero padded. As later explained in this chapter, this helps the agent stay away from the borders. Moreover, due to the lack of publicly available labeled data with varying degrees of rotation, all images are randomly rotated. Accordingly, all corpora are magnified over a magnitude of five, i.e. every image is randomly rotated four times. Let θ denote the rotation angle in radians, rotation of an image is done by applying a transformation matrix M:

$$M = \begin{bmatrix} \alpha & \beta & (1-\alpha) \cdot center.x - \beta \cdot center.y \\ -\beta & \alpha & \beta \cdot center.x + (1-\alpha) \cdot center.y \end{bmatrix} \tag{6.1}$$

where $\alpha = scale \cdot \cos\theta$, $\beta = scale \cdot \sin\theta$ and θ is sampled uniformly at random, and $-45 < \theta < 45$. Note that the last column of M is only relevant when a different center of rotation, other than $(0, 0)$, is required, which is not the case in this work. However, M is not simplified for consistency.

When performing an affine transformation on an image, the result is a change in position of facial features. Consequently, the manually set coordinates for the eye positions in the BioID corpus will change after the images are rotated. This is

Figure 6.1: Sample rotated images from the BioID corpus. Left to right, rotations at: $34, 18, 0, -21, -38$. Middle image is the original image.

not an issue for the SCAE model, which is only trained for illumination invariant feature extraction. However, the face detection agent relies on these coordinates to learn. Therefore, these coordinates need to be estimated for the rotated images in the BioID corpus.

Since the original x and y coordinates for both eyes and the chin are known, as well as the degree of rotation, the new coordinates can be estimated by:

$$\begin{aligned} y' &= y * cos\theta + x * sin\theta \\ x' &= -y * sin\theta + x * cos\theta \end{aligned} \tag{6.2}$$

Note that estimating the new coordinates is only possible due to the way the images are rotated, using an affine transformation with known center of rotation: $center.x = 0$ and $center.y = 0$ in Equation 6.1.

6.4 Unsupervised Feature Extraction

As discussed in the next section, the Deep Q-learning agent only looks at a slice of an input image at a time. Moreover, because the agent is scale invariant, the slice can have varying dimensions. Since deep NNs learn a weight matrix of fixed size, the input to the network has to always be of the same size. Therefore, the image slices are scaled using bicubic interpolation. Scaling is done using bicubic interpolation [86] as it retains more details than other commonly used methods such as bilinear interpolation. The richer quality of the resulting interpolated image is the result of considering 4×4 pixel neighborhoods to estimate the new intensity value for every

point (x, y). This is obtained by convolving the image with the kernel k:

$$k(x) = \begin{cases} (a+2)|x|^3 - (a+3)|x|^2 + 1 & , \quad \text{if} \quad x \leq 1 \\ a|x|^3 - 5a|x|^2 + 8a|x| - 4a & , \quad \text{if} \quad 1 < x < 2 \\ 0 & , \quad otherwise \end{cases} \tag{6.3}$$

Accordingly, the SCAE model is trained on random crops of the images in the MultiFaces corpus. During training, a set of random coordinates C is created along with a random dimension d which is equal to or smaller than the smaller dimension of the images in the training data: 286 in this case. Then an image is sampled from the training set and cropped according to C and d. The resulting cropped image is scaled to 200×200 and becomes x. The same process is done for its corresponding target image x_μ. This is done at run time in order to have a larger number of possible crops and cover as much area of an image as possible.

The SCAE model is trained following the illumination invariant Gradual-GLW training method from Chapter 4, where the input image x is mapped to a hidden representation $h = f(x)$ and back to a reconstruction y which is an approximation of x_μ and x_μ is the image with good luminance. For the images from the MultiPie and Yale corpora, x_μ is the image with relative luminance Y closes to the mean. For the remaining corpora, x_μ is the original image before gamma correction.

The SCAE model is trained until the stopping criteria is met: once the reconstructed images have a similar relative luminance. Once training is complete, the decoder element is removed and the encoder is used as the feature extraction method for DeepFace. Note that there are other feature extraction methods such as PCA which can be used for dimensionality reduction, and no pre-processing is actually required other than scaling since the Deep Q-Network can take any input. However, because the objective is to have a scale, translation, and illumination invariant face detector, the SCAE model greatly assists in achieving this goal. Moreover, unlike PCA, the SCAE model retains the spatial structure of the input image which plays a significant role in face detection.

As opposed to the model in Chapter 4, the SCAE model here only employs three convolutional layers and no max pooling. Pooling layers are avoided in an attempt to keep the structure of the input as intact as possible. And instead, down sampling is done using a stride of 2, i.e. moving the filter kernel two places instead of one, in some of the convolutional layers.

6.5 Deep Q-Learning Face Detection

Recall from Section 2.10 in the literature Chapter 2. In order to use DRL as a learning concept for face detection, it is necessary to formulate the task as a Markov Decision Process (MDP). MDP models are defined by a finite set of possible world or environment states, a finite set of possible actions that the agent can execute at any given time, a function describing the probability transition from one state to the other after executing an action, and a reward function describing the effect of executing a given action in a given state. To qualify as an MDP, a Reinforcement Learning (RL) task has to satisfy the Markov property, which in effect means that a probability distribution of future events at time $t + 1$ depends solely upon the environment's state at time t and not on the events before t [49]. To comply with this rule, DeepFace is based on stochastic transitions, and at any point in time t, the probability that the agent will reach its desired destination depends solely upon the current state of the environment.

The face detection DRL model employs Deep Q-Learning (DQL) to find an optimal action-value policy that encourages an agent to navigate towards a face within the image space. Similarly to Q-Learning, DQL is an off-policy Temporal Difference —meaning it is able to learn from raw experience without a model of the environment's dynamics—based algorithm that utilizes a deep neural network as function approximation to minimize temporal difference lose. In this case, the Deep Q neural network (Q-network) is a CNN composed of four convolutional layers. As opposed to conventional deep CNN models, the Q-network here does not utilize max pooling layers in order to retain as much spatial information as possible. Whereas max pooling

does not lose much spatial information due to the small kernels used, convolutional layers with a stride greater than one are better at retaining spatial information as well as structural information, and can also downsample a feature vector in the same manner as pooling layers. Formally, DQL is defined by [87]:

$$Q^*(s,a) = \max_{\pi} \mathbb{E}[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + ...|s_t = s, a_t = a, \pi] \tag{6.4}$$

where r_t is the sum of discounted rewards at time step t, with a discount factor γ determining the agent's horizon. This sum of rewards is also shaped by policy π:

$$\pi = P(a|s) \tag{6.5}$$

where s denotes an observation of the environment, that is the image space occupied by the bounding box, and a the action taken by the agent.

Figure 6.2: Visualization of the DRL face detection model. Red bounding box denotes the agents random starting position. Landmarks marked with X are the ones used to determine the desired target: in between the blue and green boxes. Agent coordinates: lower left corner $(x_1, y_1$, upper left corner (x_2, y_2), lower right corner (x_3, y_3), upper right corner (x_4, y_4).

In this task of face detection, the agent's job is to place a bounding box around three landmarks, namely coordinates, describing a face. Given that each facial expression image in the BioID database is labeled with a set of manually set coordinates

describing the subject's chin and eyes, these three coordinates are used as target for the agent. As illustrated in Figure 6.2, the agent has some freedom on how close or how far to place the bounding box from the target coordinates.

The bounding box is initially randomly placed within image space and set to a random size that covers at least 50%, or up to 80%, of the image and. This facilitates the job of the agent to cover as much ground as possible and get an idea of where it is relative to the face. Smaller initial sizes for the bounding box have been observed to lead the agent to shift the bounding box away from the target given that it has no information of where it is located relative to the image. Although the learning process would be easier if the bounding box is initially placed at a fixed location, experiments showed that on average, a random initial position leads to faster localization.

During each episode, the area covered by the bounding box is cropped from the image, resized to 200×200 and forward propagated through the illumination invariant SCAE model. The resulting feature planes are then passed through the Q-network, which outputs the action the agent should take. Note that this is only the case when ϵ determines that the action should be selected by the Q-network and not be random as described below.

The possible actions a an agent can execute are: move up, move down, move right, move left, shrink the bounding box, enlarge the bounding box, rotate left, rotate right, or come to a stop, thus formally $a \in \mathbb{Z}_0 : a \leq 8$. Every time the agent decides to transform the bounding box, this is done by a adding or removing five pixels over the y dimension. In contrast, when the agent decides to move the box, it is done by moving the bounding box 10 pixel values in a given direction. If the action selected is to rotate the bounding box, it is rotated according to Equation 6.1 and $\theta = 10$ if rotating left, or $\theta = -10$ if rotating right. This combination seems to provide the right trade-off between speed and accuracy. Larger values have shown to make the agent misplace the bounding box or miss the target by a small number of pixels, falling in an infinite loop trying to position the bounding box over the face.

The environment's state s at any given time step t is denoted by a list describing: the size of the bounding box, the (x, y) coordinates describing the position of the bounding box in relation to the image, and the height, width and angle of the bounding box. Consequently, the state s_{t+1} is defined by the action a_t executed during the state s_t, as illustrated in Table 6.1. When the action selected by the Q-agent is to stop, the episode concludes and list describing the bounding box at state s_{t+1} is returned.

As illustrated in Table 6.2, the objective of the agent is to place the bounding box around the desired coordinates described by the eyes and chin. Therefore, the target T describes these landmarks $T = \{L, R, C\}$ where L and R denote the (x, y) coordinates for the left and right eyes, and C the coordinates describing the chin. After every action is executed, the coordinates describing the top left corner of the bounding box (x_3, y_3) are compared to L, the right top coordinates of the bounding box are compared to R, and the bottom two coordinates (x_4, y_4) are compared to C. Comparison is done using the euclidean distance from one point to another. The resulting three distances are then combined and also kept in memory as d_t.

If the bounding box covers all three landmarks in T and is within a threshold distance, the flag *at_target* is set to *true*, and the agent may be rewarded according to the following criteria:

$$r_t = \begin{cases} +10 & , \quad \text{if} \quad a_t = 8 \quad \& \quad at\_target \\ +1 & , \quad \text{if} \quad d_{t+1} < d_t \\ -10 & , \quad \text{if} \quad terminated \\ -1 & , \quad otherwise \end{cases} \tag{6.6}$$

where d_{t+1} denotes the new accumulated distance from the target, and *terminated* is a flag indicating if the episode was terminated early, e.g. if the box went out of image space or the bounding box became too small. In this reward function, if the agent selects an action that places the bounding box further away from the target, it gets punished. If it goes out of bounds, it gets punished more. If the agent selects an action that brings it closer to the desired target, it gets some reward. And finally, if it ends the episode at the desired target, it gets rewarded more.

Table 6.1: Transformations to the bounding box: $s_{t=1} : (a_t, s_t)$

f is a function that rotates an image using Equation 6.1 and returns the new estimated angle of the image α. ℓ is a function that returns updated (x, y) coordinates for a given point using Equation 6.2. w and h denote the width and height of the bounding box.

Action	**Transformation**
$a_t = 0$ move up	$y_1 = y_1 + 10$ $y_2 = y_2 + 10$ $y_3 = y_3 + 10$ $y_4 = y_4 + 10$
$a_t = 1$ move down	$y_1 = y_1 - 10$ $y_2 = y_2 - 10$ $y_3 = y_3 - 10$ $y_4 = y_4 - 10$
$a_t = 2$ move left	$x_1 = x_1 - 10$ $x_2 = x_2 - 10$ $x_3 = x_3 - 10$ $x_4 = x_4 - 10$
$a_t = 3$ move right	$x_1 = x_1 + 10$ $x_2 = x_2 + 10$ $x_3 = x_3 + 10$ $x_4 = x_4 + 10$
$a_t = 4$ enlarge	$x_1 = x_1 - 5, \quad y_1 = y_1 - 5$ $x_2 = x_2 - 5, \quad y_2 = y_2 - 5 \quad w = \sqrt{(x_1 - x_2)^2 + (y_2 - y_2)^2}$ $x_3 = x_3 + 5, \quad y_3 = y_3 + 5$ $x_4 = x_4 + 5, \quad y_4 = y_4 + 5 \quad h = \sqrt{(x_3 - x_1)^2 + (y_3 - y_1)^2}$
$a_t = 5$ reduce	$x_1 = x_1 + 5, \quad y_1 = y_1 + 5$ $x_2 = x_2 - 5, \quad y_2 = y_2 + 5 \quad w = \sqrt{(x_1 - x_2)^2 + (y_2 - y_2)^2}$ $x_3 = x_3 + 5, \quad y_3 = y_3 - 5$ $x_4 = x_4 - 5, \quad y_4 = y_4 - 5 \quad h = \sqrt{(x_3 - x_1)^2 + (y_3 - y_1)^2}$
$a_t = 6$ rotate left	$\alpha_t \leftarrow f(\theta = 10)$ $x_1, y_1 \leftarrow \ell(x_1, y_1)$ $x_2, y_2 \leftarrow \ell(x_2, y_2)$ $x_3, y_3 \leftarrow \ell(x_3, y_3)$ $x_4, y_4 \leftarrow \ell(x_4, y_4)$
$a_t = 7$ rotate right	$\alpha_t \leftarrow f(\theta = -10)$ $x_1, y_1 \leftarrow \ell(x_1, y_1)$ $x_2, y_2 \leftarrow \ell(x_2, y_2)$ $x_3, y_3 \leftarrow \ell(x_3, y_3)$ $x_4, y_4 \leftarrow \ell(x_4, y_4)$
$a_t = 8$ stop	return $\quad s_{t+1} = \{x_1, \ldots, x_4, y_1, \ldots, y_4, \alpha, w, h\}$

The DRL model also employs experience replay as done by [88], by storing the last N experience $e_t = s_t, a_t, r_t, s_{t+1}$ tuples in replay memory D and then sampling uniformly at random from $D_t = e_1, ..., e_t$, when performing updates in order to encode past actions. Note that the cropped bounding boxes are the ones saved in memory, instead of list describing it. This process is formally described by:

$$L_i(\theta_i) = \mathbb{E}_{(s,a,r,s')\sim U(D)}\left[\left(r + \gamma \max_{a'} Q(s', a'; \theta_i^-) - Q(s, a; \theta_i\right)^2\right] \tag{6.7}$$

where θ_i denotes the parameters of the Q-network at iteration i, Q_i^- denote the Q-network parameters used to compute the target at iteration i, and $(s, a, r, s') \sim U(D)$ are the mini-batches sampled from D.

The agent follows an ϵ–*greedy* strategy to provide the right balance between exploration and exploitation. Initially, ϵ is set to 0.9, i.e. explore 90% of the time by picking random actions, to allow for more exploration in early episodes. This is then annealed linearly to 0.1 over the first 1000 iterations and fixed at 0.1 thereafter. 1000 iterations have proved to be enough for the agent to explore and learn to pick actions that provide more reward. Once the Q-network starts learning to select informed actions on its own, it is only allowed to pick a random action 10% of the time, providing the right balance between informed decision making and exploration.

During training, the agent is initially allowed to try for a maximum of 120 attempts. Once ϵ is set to 0.1, the agent is only allowed to try for a maximum of 50 attempts. If the agent does not find the face during this episode, the *terminated* flag is set to *true* and, thus, the reward for the last time step t becomes negative according to Equation 6.6. The *terminated* flag is also raised if the agent rotates the bounding box for more than 50 degrees in either direction.

To discourage the agent from going out of image space, it is punished if it goes out of bounds. Moreover, the image is zero-padded with a large margin in order to let the agent know that if it reaches this hard line it is shifting away from the target and should travel in a different direction. When zero padding the image, the

set coordinates for the landmarks of interest change. However, they can be easily estimated by adding or subtracting the number of zero-padding pixels to the x and y coordinates.

The Q-network is trained for $175,000$ episodes using a replay memory of size $10,000$. Learning rate is set to 0.6 and annealed using a rate of 0.01 using Equation 3.16 until it reaches 0.00001. Training is done using SGD with Nesterov momentum of 0.9 and mini-batches of size 64.

6.6 Face Detection Results and Discussion

Once training concludes, DeepFace is evaluated on the testing subset of the BioID corpus and the 100—note that because these images were also randomly rotated, they are now 500 in total—images randomly selected from the Multi-Pie corpus. The latter were randomly selected to evaluate the model on novel, i.e. that are not part of the same corpus used for training, images with varying luminance. DeepFace achieves an accuracy rate of 96.53% on the testing subset of the BioID, and 93.60% on the Multi-Pie subset. Table 6.2 illustrates the results on both corpora.

Table 6.2: Face recognition results on the BioID and a subset of the Multi-Pie corpora.

	$BioID$	$MultiPie$
$Success$	2201/2280	468/500
$Fail$	79/2280	32/500
$Total$	96.53%	93.60%

As seen in Table 6.2, the face detection model classifies 2201 images correctly out of 2280 in the test subset of the BioID corpus. It is worth noting that out of these, DeepFace had a 100% success rate on the 456 non-rotated images. On the Multi-Pie corpus, DeepFace had a success rate of 93.60%. Out of the 100 non-rotated images, a face was successfully identified in 98 of them.

During testing, the average detection rate on the BioID was 17 transitions from state s' to s, i.e. the agent performed 17 actions to placed the bounding box around the face. On the Multi-Pie, the average was 32. The difference can be justified by the fact that faces in the Multi-Pie corpus are smaller, and also the Q-network has never seen these images. This also explains the lower performance on this corpus.

For the randomly rotated images, DeepFace returns a non-rotated image upon successful localization. Since the bounding box is always placed without any rotation, i.e. x_1 and x_3 have the same value initially, and since when the Q-network decides to rotate the bounding box this is kept in the replay memory, it is possible to know exactly how much the bounding box was rotated before reaching its target. Then upon recognition, DeepFace rotates the image using Equation 6.1 in the opposite direction. Figure 6.3 illustrates a pictorial description of this process.

Figure 6.3: Rotation invariant face localization.

In some cases where the face detector failed, it was observed that in many cases it actually placed the bounding box within the target. However, the Q-network never selected to end the episode, i.e. $\alpha = 8$, and, therefore, it was called a failure. This is particularly important since in a real life scenario, where there is no labeled data, there is no way to efficiently and manually tell the agent that it has reached the target and should therefore end the episode.

One of the main advantages of this method is that it is illumination invariant. The model did not struggle to deal with illumination, and the majority of the failures happened due to other issues such as rotation. For example, in some instances, the

agent either rotate the bounding box too much, or got stuck in the same position by selected a transformation action and then undoing it in the next time step. In some other instances, the bonding box became too small, rendering the process a failure.

One observation made during early episodes, with a high ϵ–*greedy* parameter is that the agent often would drive the bounding box out of image space. This can be justified by the lack of information and knowledge in early episodes in which the agent has to explore more in order to accumulate knowledge. Moreover, the agent does not have any information of where the image ends or starts. This issue was solved by zero padding the images. In order to allow the agent to correct its path if it reaches the border, the padding covers at least as many pixels as the agent is allowed to move the box.

When trying other reward criteria, the agent learned but did not perform as well, or took many more steps to reach its target. It was also observed that bigger step sizes, e.g. letting the agent move the box more than 10 pixels at a time, the agent would often miss the target, or would reach it but struggle placing it within the given margin. In contrast, the smaller the step size, the better the performance but the longer training required and the longer time it took for the agent to reach the target. As a result, using a step size of 10 seemed to provide the right trade off between speed and performance.

Memory replay did not seem to affect the learning much. For instance, using a replay buffer of $20,000$ did not make much of a difference. This may be due to the small size of the dataset used to train the Q-network. In addition, letting the agent try for more than 120 attempts in early episodes did not impact the learning. Most of the time the episode would conclude before reaching 120 attempts due to: the bounding box becoming too small or to large, or ϵ determining that the random action is to stop. It is worth noting that letting ϵ randomly stop the episode played an essential role in the learning process. Otherwise, the Q-network would not select this action since it did not exist in the replay memory.

The main restrictions of DeepFace are that it was designed to work on images with a rotation $\theta : -45 < \theta < 45$. Future work could consider letting the agent have the freedom to rotate the bounding box at any desired number of degrees. Moreover, future work could explore letting the agent grow in only one direction.

6.7 Comparison Against State-of-the-art

The authors of [51] employ a similar method for face detection. They propose using PCA for dimensionality reduction and use an MLP for face detection, where they use the midpoint between the eyes as their desired target. Using distance as performance metric, the authors obtain 40.11% on the BioID corpus when their metric is 10 pixels within the target, i.e. finishing within ten pixels from the target, and their best performance is 99.62% when using 50 pixels. They compare their method to the Viola-Jones [82] and obtain 89.64% and 90.02% when measuring using 10 and 50 pixels, accordingly. Although the Viola-Jones detector obtains 90.02% at 20,30, and 40 pixels.

In contrast, DeepFace achieves 100% on non-rotated images, and an overall of 96.53% when considering rotate images. This metric is based on a 10 pixels within the desired target. For comparison, [51] obtain 40.11% at the same distance, or 79.85% at 20 pixels. The authors also report 89.64% and 90.02% using the Viola-Jones detector, at 10 and 20 pixels from the target. Demonstrating that DeepFace provides better generalization performance than empirical methods such as the Viola-Jones [82] detector, or similar methods also relying on Deep Q-Learning [51].

One of the main advantages of DeepFace is that it is guaranteed to cover the entire face, while at the same time ignoring background noise. Whereas the method by [51] is not aware of the size of the face, and therefore, would not be able to crop it appropriately. Another advantage of DeepFace, is the fact that it is rotation and illumination invariant, and solely relies on CNNs for feature extraction and feature learning.

Other work on face detection using RL is done by [50]. However, the authors only use RL to find dominant features in every image of the training data. Similarly [52] also try to address illumination invariance by employing γ correction and Deep Q-Learning. However, the authors focus on person identification through facial recognition, rather than face detection.

Similarly, [54] employ deep RL for face recognition under different levels of illumination. The authors also employ gamma correction to train and test their model under different levels of image luminance and obtain 100% precision scores when $\gamma \in \{0.5, ..., 1.6\}$. Although the results obtained by the authors are remarkable, the method proposed here can deal with more variations in image luminance: $\gamma \in \{0.4, ..., 3.4\}$, and is also rotation invariant.

Although not considered in this work, other state-of-the-art face recognition approaches have been proposed by [89]. The authors employ a Faster R-CNN [90], which are the current state-of-the-art models in object recognition, model for group face recognition in unconstrained environments. Similarly, [63] have proposed a state-of-the-art pose invariant face detection model.

6.8 Chapter Conclusion

This chapter of the thesis has introduced a rotation and illumination invariant DRL face recognition model. The face detection model learns using temporal difference learning by using previous experiences to predict future events. The development of this model was inspired by the observation that empirical face detection models are prone to failure on images with nonuniform conditions, for instance on images with very low relative luminance. Moreover, because face rotation is widely ignored by state-of-the-art detectors, it was taken into consideration in this chapter.

DeepFace was demonstrated to achieve state-of-the-art recognition rate on faces with some degree of rotation. And was demonstrated to be robust to changes in

illumination. The success of DeepFace partially relies on the illumination invariant model from Chapter 4, which is employed for feature extraction. Although DeepFace has some limitations, e.g. it was not evaluated on multi-pose images or on group detection, it demonstrates the potential of the experimental design and should not require major adjustments to deal with other forms of invariance.

This chapter compliments existing work in the domain of face recognition by proposing a robust rotation and illumination invariant learning algorithm. Future work can investigate extending the model to deal with pose invariance as well as group face recognition.

Chapter 7

Conclusion

7.1 Introduction

This thesis has explored the development of deep and deep reinforcement learning models for face detection and emotion recognition from facial expressions. More precisely, the main objective of this research was to investigate and provide an answer to the following research question:

"Is it possible to develop novel artificial neural network architectures based on deep and reinforcement learning concepts to efficiently recognize faces and human emotions through facial expressions in unconstrained environments?"

The inspiration of this research is the significant role played by human emotions in daily life, as well as the importance of being able to correctly perceive and interpret emotions in other people. Although several works using machine learning methods, and deep learning in particular, have been proposed to address automated emotion recognition, they do not address some of the main challenges in automated emotion recognition: generalization on nonuniform data. Accordingly, several novel deep architectures and learning methods were designed with emphasis on pose and illumination invariance, as well as network complexity.

Existing work in the domain of emotion recognition is commonly done by analyzing

a person's facial expressions [13], speech signals [91], body language [92], or other physiological information such as EEG signals [93]. Whereas detecting emotions can be done using all these affective modalities, whether combined or individually, some of them are difficult to obtain. For instance, obtaining physiological information is rather intrusive and usually requires physical contact, e.g. an EEG or heartbeat scanner. Similarly, speech signals are often mixed with background noise, and body language is difficult to capture in an adequate manner. In contrast, facial expression images are easier to obtain, are non-intrusive, and have proven to be efficient for emotion recognition. Consequently, the work presented in Chapters 3–5 employs facial expression images. Nonetheless, facial expression images are also subject to changes in illumination and facial pose, cultural differences, among others, all of which increase the difficulty of recognizing emotions in an efficient manner. Therefore, Chapters 4 and 5 focused solely on overcoming illumination and pose invariance, as they are arguably the biggest challenges in automated emotion recognition from faces to be solved.

The work in Chapter 6 was inspired by observations made during the design of the DL models for emotion recognition from facial expressions. Because face detection is the first image pre-processing step, if the face detector fails to find a face on a given image, the task of recognizing emotions can become unattainable. Moreover, some of the best performing face detector models [63] are very computationally expensive and are not suitable for real time recognition due to their large latency.

This thesis was designed to investigate the possibility of overcoming such limitations of contemporary face and emotion recognition models. The results are a variety of novel deep learning architectures and learning paradigms for emotion recognition, and a novel architecture for face detection. As a whole, these contributions address face and facial emotion recognition in unconstrained environments. All results in Chapters 3–6 are reported as an average of ten experimental runs.

7.2 Thesis Contributions

The originality and scientific value of this thesis is presented in the form of deep learning methods for emotion recognition as presented in Chapters 3,4, and 5, and the deep reinforcement learning methods for face detection, as presented in Chapter 6. The main contributions can be summarized as:

- An illumination invariant Stacked Convolutional Autoencoder model capable of reconstructing images with up to 64 different degrees of illumination as images with the same illumination.
- A Gradual Greedy Layer-Wise training algorithm that reduces error accumulation in early layers and significantly improves reconstruction performance and training time.
- A pose invariant Generative Adversarial Stacked Convolutional Autoencoder model that can reduce face pose to zero degrees from up to 60 degrees.
- Two convolutional layers: one which utilizes $shifting$ neurons, and another one that exploits facial symmetry to reduce its number of parameters.
- Several deep CNN models that achieve state-of-the-art classification rates on data with nonuniform conditions.
- A novel deep reinforcement learning architecture designed for illumination and pose invariant face recognition.

These contributions are supported through extensive experimentation and discussion. Furthermore, the novelty of this work is supported through other contributions such as: the combination of adversarial learning and greedy layer-wise training into a single learning paradigm; a deep CNN that simplifies feature learning by splitting the input image in half and learning to extract features through two learning streams; the concept of pretraining all convolutional layers in a CNN as a shallow autoencoder regardless of the number of filter kernels used; new knowledge in the usage of rectifier linear unit functions and their effect on regression and classification problems; a deep

reinforcement learning reward function carefully designed for face detection, among others.

7.3 Deep Learning for Emotion Recognition

The thesis began by exploring two deep learning concepts and their suitability for emotion recognition: deep convolutional networks and unsupervised pretraining. Chapter 3 introduced a novel deep CNN that splits the input image in half and learns to self-extract salient features in parallel using two sub-networks. The resulting feature vector is then concatenated and used for classification. This architecture is referred to as Convolutional Ensembles Network, or CEN.

The main advantage offered by the CEN model is the simplification of feature learning, at the expense of marginally increased computational cost. Because the image is split in half and faces in the KDEF corpus are centered with a grid, it can be assumed that specific facial features, such as the eyes and eyebrows always lie within the upper half of an image, and the mouth within the lower half. By having an ensemble solely dedicated to learning salient features that resemble a mouth, or the eyes and eyebrows for the second ensemble, the sub-networks learn specific features instead of generic or broader ones. However, in terms of implementation, having two sub-networks requires two separate weight matrices and more memory. This results in marginally increased computational cost. And because the same error is equally propagated through both sub-networks, if one of the is struggling to learn it will affect the other, resulting in longer training times.

The CEN model obtained a test accuracy rate of 86.73% on the KDEF, after training for 5280 epochs. Although these results fall behind the state-of-the-art, they prove that the concept of splitting images in half and taking advantage of locality is a good training approach. Although, it requires more exploration, it was not used in later architectures due to the already high number of learning parameters. However, during training, the CEN model also helped in noticing some of the challenges

in training deep CNN. For instance, using sigmoid activations instead of ReLUs resulted in lower classification performance and often in vanishing gradients. sigmoid activations also demonstrated to be more prone to the way the hyperparameters were initialized, for instance a higher learning rate would result in exploding gradients. All these observations greatly assisted in the training of the other architectures in this thesis.

Chapter 3 also explored the concept of pretraining a deep CNN as a stacked convolutional autoencoder. The observations gather in the training of the CEN model, along with existing literature against random weight initialization, inspired this architecture. This model employs ReLU activations instead of sigmoid.

The main challenge addressed by this model was the reconstruction of high dimensional feature planes produced by convolutional layers. Standard autoencoders composed of MLPs only produce a one dimensional feature vector and, therefore, are not as difficult to train. However, because convolutional layers normally employ many filters —it is common to increase the number of filters in every layer as the network grows—they produce a feature vector composed of many feature planes. For this reason, it is common to only pretrain the first layer of a deep CNN as an autoencoder, since the autoencoder only has to reconstruct one input plane, or three if using colored images. However, as the CNN grows, the more feature planes an autoencoder has to reconstruct. Accordingly, the SCAE model introduced the application of batch normalization [39] to speed up training, eliminate the need for dropout, and eliminate the risk of vanishing or exploding gradients.

The SCAE model also showed that fine-tuning the final stack of shallow autoencoders also improved the reconstruction error of the model. When the SCAE completed training, the encoder element was used to initialize a CNN classifier, which was fine-tuned for only 20 epochs. The deep CNN achieved an accuracy rate of 92.52%, compared to when 91.16%, when the CNN was initialized with a random distribution and trained for 500 epochs.

The unsupervised pretraining of the CNN as a SCAE provided three main findings:(i) large convolutional models can in fact be pretrained in a GLW unsupervised fashion, (ii) unsupervised pretraining leads to better performance of a deep CNN, and (iii) the greedy layer-wise training method has one vulnerability, namely error accumulation. These findings formed the basis of the pose and illumination invariant architectures presented in the remaining chapters.

It was also observed in this chapter that the filters learned by the first convolutional layer in a deep CNN resemble Gabor filters [40] and learn generic features.

7.4 Illumination Invariant Emotion Recognition

When analyzing the results obtained in Chapter 3, it was observed that most of the missclassified images by the CEN and the CNN model pretrained as SCAE had something in common: they appeared significantly brighter or darker, as opposed to the remaining images in the training and testing data. Accordingly, Chapter 4 investigated this issue further and introduced an emotion recognition model designed to address illumination invariance.

The illumination invariant model in Chapter 4 also employs a SCAE to pretrain a deep CNN. However, the SCAE model is trained to learn an illumination invariant feature vector. The illumination invariant SCAE learns a function $f(x)$ that produces a downsampled illumination invariant feature vector h, which is mapped to a reconstruction y that resembles the target x_μ. Therefore, instead of simply learning an identity function, like empirical autoencoders and the SCAE model from Chapter 3, it learns $g(f(x)) = x_\mu$. In this case x_μ, is a copy of x with equal or different relative luminance levels, thus $\neg\Box(x = x_\mu)$.

The illumination invariant SCAE model was demonstrated to generalize on novel data, e.g. from different corpora than the one used during training, and produce

remarkable illumination invariant reconstruction. The model also managed to reconstruct very dark images in which a face is not very visible. The main limitation was observed to be its dependability on multi-illumination data: In order to map an input image with some arbitrary degree of illumination, to a second image with a fixed level of illumination, both images have to exist in the training corpus. However, it was demonstrated that when multi-illumination corpora is nonexistent, gamma correction can be applied to a corpus to create variations of a given image with several levels of illumination. Moreover, applying gamma correction to a corpus also magnifies the training data, which is usually beneficial for DL models.

Because the feature vector h produced by the SCAE is illumination invariant, the distribution of $q(h)$ is significantly smaller than that of the input data $p(x)$. In effect, when the encoder element of the SCAE is used to initialize a CNN, the CNN does not have to learn multi-illumination representations and instead only deals with one degree of illumination. This results in faster learning and significantly better generalization of the CNN. Accordingly, when this CNN was evaluated on the KDEF corpus, it achieved an accuracy rate of 95.70%, compared to 92.52% achieved by the non-illumination invariant model from Chapter 3. Similarly, when evaluated on the CK+ corpus, it achieved an accuracy rate of 94.90%, whereas when a CNN was trained without the illumination invariant method proposed it only achieved 86%. In addition, when the illumination invariant deep CNN was evaluated on the CFE corpus, which is made up of the CK+, KDEF, JAFFE, and FEEDTUM corpora, it achieve a state-of-the-art classification rate of 99.14%. Note that the results obtained on just the KDEF corpus are also considered state-of-the-art and are only surpassed by the pose invariant model discussed in the next section.

These results strongly support the illumination invariant training approach proposed in Chapter 4. Nonetheless, two important key factors played a significant role in both, the reconstruction and classification results: ReLU-n activation functions and gradual greedy layer-wise training. ReLU-n activation layers were initially introduced to assist in the learning of an illumination invariant feature vector by setting a threshold on the activations of a given node. However, they were also shown to

have an effect in classification, and that a lower threshold marginally greater than zero improves classification performance. In contrast, Gradual-GLW was introduced to address the vulnerability of GLW to error accumulation.

Gradual-GLW improves training of SCAE models by reducing error accumulation in early layers, and stopping it from being propagated to deeper layers. This is achieved by fine-tuning the stack of autoencoders at every step $k \in \mathbb{Z} : k \in [1, m]$ using raw pixel data. When adapted for the the illumination invariant model, the stopping condition in Gradual-GLW unsupervised training is done according to the difference between the estimated luminance of the reconstructed images and that of the target image.

7.5 Pose Invariant Emotion Recognition

One of the major challenges in the domain of emotion recognition from facial expressions is dealing with changes in facial pose. To the best knowledge of the authors, no existing work in the literature has attempted to address this issue specifically. Chapter 5 introduced a novel deep learning method designed solely to address pose invariance. The pose invariant model is based on a novel generative adversarial autoencoder (GASCA) architecture trained using Gradual-GLW.

Similarly to the illumination invariant model from Chapter 4, the GASCA model learns a feature vector by mapping the input x_φ to a hidden representation and back to a reconstruction y that resembles the target image x_μ. And x_μ and x_φ are both images belonging to the same subject but taken from different angles. Learning is done using a version of Gradual-GLW adapted for adversarial learning and, thus, a discriminator network D is created. Empirical adversarial autoencoders draw samples from a random distribution, and use them along with the code vector of produced by the encoder function in G as input to D. In the GASCA model, D gets the reconstruction y produced by G and tries to differentiate it from the target image x_μ. In contrast, G is trained to trick G into believing that the reconstructions are the same

as the target images. Which in effect forces G to produce remarkable reconstructions. Moreover, D is also trained in a layer-wise fashion. Both D and G are also fine-tuned as they grow, i.e. when more layers are added.

The GASCA model is able to reduce facial pose from up to ± 60 degrees down to 0 degrees and produces remarkable reconstructions. Remarkable reconstructions were obtained even in cases where half of the face is not visible, yet the model managed to compensate for the missing information and even retain the shape of facial features important for emotion recognition.

The encoder element of the GASCA model is used to initialize a CNN and fine-tune for classification. Effectively, the SCAE model reduces the data distribution and leads to faster fine-tuning of the CNN models. The CNN models achieves a classification performance of 96.81% on the KDEF corpus. Note that this version of the KDEF has facial expression images with estimate poses of up to ± 45 degrees. When trained for pose and illumination invariance, the model achieves state-of-the-art classification performance of 98.07%. Moreover, when tested on nonuniform data from 28 participants collected using a NAO robot in unconstrained environments, it achieves an accuracy rate of 81.36%, compared to 73.55% when trained only for illumination invariance, as we reported in [78].

Part of the outstanding performance of the GASCA model is attributed to the two new convolutional layers designed specifically to assist the model in learning a pose invariant feature vector. The ConvMLP layers are convolutional layers that employ $shifting$ neurons to allow reduction of facial pose in faces. HalfConv layers are layers designed to exploit facial symmetry and reduce the number of parameters in half by only processing half of an image and then mirroring its output.

The GASCA model, when trained to address illumination and pose invariance, eliminates the need for more complex image pre-processing steps often found in the literature: noise injection, color and brightness normalization, among others [14]. This is particularly important considering that these pre-processing methods often

lead to an increase in the data distribution space.

7.6 Illumination and Rotation Invariant Face Detection

Although face recognition is a widely studied subject in the visual processing and ML communities, work addressing face rotation is very limited. Chapter 6 introduced a novel DRL architecture for illumination and rotation invariant face detection. This model achieved state-of-the-art recognition rate of 96.53% on the testing subset of the BioID corpus, which contains randomly rotate images. The model also provides good generalization performance on novel data and achieved a performance rate of 93.60% on unseen images of the Multi-Pie corpus.

DeepFace provides an alternative to popular face detectors such as the Viola-Jones [82], or Histogram of Oriented Gradients [84], by employing a non-exhaustive search method. Moreover, DeepFace returns a non-rotated cropped face. Whereas other methods would not be able to achieve this without employing alternative rotation estimation methods.

Although not explored in this thesis, in theory, DeepFace should work on multi-pose images by either adding multi-pose images and processing them as is, or by employing the pose invariant GASCA model from Chapter 5 and using the pose invariant feature vector to train the deep Q-network model. Either approach should provide an alternative to contemporary state-of-the-art pose invariant methods such as the one proposed by [63]. For instance, once loaded into memory, a single image is processed in 1/100th of a second on average, compared to almost 45 seconds required by the best model proposed by [63]. Note that these comparison is done based on the multi-core open source code released by the authors; nonetheless, DeepFace should be much faster as it only requires some matrix multiplications.

The robustness of DeepFace on images with varying illumination is attributed

to the illumination invariant SCAE model from Chapter 4 trained using Gradual-GLW and γ corrected images. Therefore, supporting the work presented in the same chapter.

7.7 Research Limitations

One of the main limitations of the work presented in this thesis is the emotion recognition models are trained on images of white Caucasian subjects. This is due to lack of existing publicly available multi-cultural facial expression data. And since people from different ethnic backgrounds express emotions in different ways [81], the emotion recognition models offer lower classification performance on images of participants from different cultures. This was observed when testing the model of the NAOFaces corpus, in which participants are from at least five different ethnic backgrounds. Although this was reduced to some degree by combining various corpora into a single one, the results on the NAOFaces corpus were still lower compared to the other corpora.

In terms of pose invariant recognition, the GASCA model is only evaluated on images with an estimated pose of up to ± 60 degrees. For facial expressions with a larger facial pose, the model's performance will likely drop. This has been observed in the reconstructed images: when an image with estimated pose at 0 degrees is passed through the GASCA model, the reconstruction is marginally better than for images with larger facial pose. And the larger the pose, the lower the quality of the reconstruction, although the difference is marginal and for the purpose of this work it is trivial. The illumination invariant model proved to be more robust to drastic changes in illumination.

Furthermore, because the emotion recognition models were trained on corpora containing only seven emotional states, they are unable to detect more complex emotions such as shame, trust, or envy, among others. In addition, because classification

is done categorically, the models are unable to deal with overlapping emotions or transitions in emotional states.

One aspect that was not considered in this research was real-time emotion recognition. Theoretically, the emotion recognition models should not have any latency issues: for a batch of 512 images, the average prediction time is one tenth of a second when processed using two NVidia Kepler K80 GPUs, and implemented using the Torch7 framework [94]. Although this does not account for face detection or the latency caused by the camera used to obtain the images.

The main limitation of the findings presented on the face detection model was the lengthy training time required to learn a policy: usually over $100,000$ iterations. Moreover, this research did not look into extreme facial rotation, for instance upside down faces. The face detection model was also evaluated on a single corpus due to its reliability on three known facial points during training. The face detection model also did not take into consideration multi-pose corpora due to lack of labeled data.

7.8 Future Direction

Automated emotion recognition is an area of research that continues to expand. Its applications are diverse and can be employed in the education sector, healthcare, security, social robotics, among many others. Future work can explore the suitability of the pose and illumination invariant emotion recognition models for real time recognition in these, and other domains.

The illumination and pose invariant models can be extended to consider more complex emotions, such as trust or envy, as well as to take into account overlapping emotions and transitions between emotions. In addition, future work can extend the findings presented in Chapters 3–5 to provide better generalization insensitive to cultural differences. This can be achieved by incorporating multi-cultural facial expression images in the training data. The emotion recognition models can also

be expanded to consider multimodal affective data. For instance, speech signal in the form of spectrograms; or, in combination with other empirical classifiers, such as MLPs, where both models learn simultaneously.

Due to the formulation of the GASCA model, and due to the success of generative adversarial learning in synthetic image generation, it could be extended to generate realistic facial expression images. This could be done using the encoder element in the generator model, or a second decoder model could be added. Some variations of this approach for generation of synthetic facial expression images has been explored by [95], [96] and [97]. Such extension would be very beneficial for the domain of emotion recognition, considering that labeled data is limited.

The formulation of the GASCA and SCAE models could also be explored to address tilt and face rotation. Although tilt is inherently fixed by the pose invariant model to some degree. For face rotation, the GASCA model could use the non-rotated image as the reconstruction target, and gradually reduce face rotation through the shallow autoencoders. Hypothetically, the GASCA model should also be able applicable to other domains, such as rotation invariant object recognition. Moreover, both the SCAE and GASCA models could be evaluated on corrupted images, where parts of the images contain no significant information. And they could be combined with sub-pixel convolutional layers [79] to improve the resolution of the reconstructed images. However, in this work the quality of the reconstructions is trivial.

Other possible extensions to the SCAE and GASCA models is end-to-end learning that takes into account classification. For instance, in the GASCA model, the latent distribution $q(z)$ could be used as input to a second classifiers—the discriminator D is a classifier model—for emotion recognition. The classification loss could then be combined with the adversarial loss to update all the models. However, this may not be straight forward considering that this would add complexity to the overall architecture. Moreover, it would have to be explored if such classifier should be trained layer-wise like the discriminator and generator models, or jointly. Nonetheless, this may be a very promising learning procedure given that in unsupervised learning, the

features learned are relevant for reconstructions but are not guaranteed to be relevant for classification.

The ConvMLP layers can also be applied to other classification and regression problems. ConvMLP layers could also be adapted for different purposes. For instance, in the state-of-the-art architectures, such as ResNets [14], the layer with $shifting$ neurons could be used to replace the skip connections. As such, instead of simply being an identity function, the skip connection would apply a function f that can be not only assist with the flow of information, but also assist in learning other complex relations. Future work can also explored increasing the depth of the fully connected layer shared across the depth dimension in ConvMLP layers.

Due to the ability of HalfConv layers to exploit facial symmetry, these could be used in other applications where symmetry is guaranteed. HalfConv layers can also be optimized to exploit locality. For instance, in cases where it is known in advance that a particular feature of interest will always be within a given region, HalfConv layers can be adapted to split the input image according to this region and learn to extra the feature of interest. In such case, depending of the application, mirroring in HalfConv layers would let the proceeding layer that this feature is important. Alternatively, instead of mirroring the extracted features, HalfConv layers can simply fill in the rest of the image with zeroes or pass the the downsampled feature plane to the next layer as is.

For the face detection model, future work should look at improving the deep reinforcement model for simultaneous recognition of multiple faces, e.g. group face detection. In theory, this could be achieved by deploying different agents on the same image, perhaps placed strategically instead of randomly. However, it may require some adaptation to avoid having all the agents find the same face. Moreover, a challenge in this scenario would be to create a strategy to determine the number of possible agents to deploy on a given image. Future work can also look at possible ways to reduce training time of the face detection model. One possible way could be to incorporate other forms of information in the training process. For instance, the

location of the agent at any given time relative to the image space.

The face detection model can also be extended to deal with multi-pose corpora. One way to accomplish this would be by using the GASCA model. For example, it could be trained to fix facial pose in full images, i.e. images with some background. In this case, once the image has a reduced pose, the agent would be able to find it, as it is, without any changes in the training of the DRL agent. Although, this would transform the image and result in a frontal face cropped image, which may not always be desired.

7.9 Chapter Conclusion

This thesis introduced a set of deep learning architectures and training paradigms designed for emotion recognition from facial expression images. Careful attention was placed on illumination and pose invariance, considering that they are two of the most challenging, and often overlooked, issues in emotion recognition from faces. Although these architectures have some limitations, the state-of-the-art classification performance they offer on nonuniform data are significant contributions to the field of automated emotion recognition. Moreover, these contributions are also significant to the field of deep learning as they introduce new deep learning concepts for invariant feature learning, as well as several novel deep learning architectures.

This thesis has also explored the development of illumination and rotation invariant face detection using deep reinforcement learning. The face detection architecture is a significant contribution to the field as it offers fast recognition offered through non-exhaustive search.

The findings presented in this research bring us a step closer to real-time emotion recognition in unconstrained environments. To the best of the authors' knowledge, this thesis is the first work designed to solely address pose invariance in emotion recognition. Along with the face detection model, the pose and illumination emotion

recognition models offer significant new knowledge in the field of automated emotion recognition, as well as the field of deep learning, and bring us a step forward in the development of intelligent systems for face and emotion recognition in unconstrained environments.

Appendix A

Supporting Material

Most of the implementations of the models in Chapters 3–6 were implemented using Torch7 [94]. More precisely, the nn, nngraph, cudnn, and autograd libraries were used as the base libraries for the implementation of all the experiments.

All experiments were ran using the several nodes with the following configurations per node: 2 Intel(R) Xeon(R) CPU E5-2683 v4, 2.10GHz (32 CPU-cores), 128GB RAM and 2 NVidia Kepler K80 GPUs.

All results are reported as an average of ten experimental runs.

The following sections describe the topology for some of the models used in Chapters 5 and 6. After various experiments, these configurations provided the best results. In some cases, some of the layers can be replaces for alternatives, for instance upsampling layers can be replaced with deconvolutional layers, with no significant changes in performance. However, these configurations provided the best performance as is.

A.1 Pose Invariant Network Topology

Table A.1: Topology of the GASCA model from Chapter 5. Left two columns are the Generator G which has two functions, an encoder and decoder. Right column describes the topology of the Discriminator D. Every row separates each shallow autoencoder, or network in D, which are trained individually. Once the GASCA model is trained, the layers in the left column are fine-tuned for classification. For the ConvMLP layers: $(filters \times filterWidth \times filterHeight, shiftingNeurons)$.

CNN/Encoder	Decoder	Discriminator
ConvMLP$(20 \times 5 \times 5, 100)$	Sigmoid	ConvMLP$(32 \times 5 \times 5, 100)$
BatchNorm	BatchNorm	BatchNorm
ReLU	Convolution$(1 \times 5 \times 5)$	ReLU
MaxPooling(2×2)	BipolarUpsampling(2)	MaxPooling(2×2)
ConvMLP$(40 \times 5 \times 5, 100)$	ReLU	ConvMLP$(64 \times 5 \times 5, 100)$
BatchNorm	BatchNorm	BatchNorm
ReLU	Convolution$(20 \times 5 \times 5)$	ReLU
MaxPooling(2×2)	BipolarUpsampling(2)	MaxPooling(2×2)
ConvMLP$(60 \times 3 \times 3, 100)$	ReLU	ConvMLP$(128 \times 3 \times 3, 100)$
BatchNorm	BatchNorm	BatchNorm
ReLU	Convolution$(40 \times 3 \times 3)$	ReLU
MaxPooling(2×2)	BipolarUpsampling(2)	MaxPooling(2×2)
ConvMLP$(80 \times 3 \times 3, 100)$	ReLU	ConvMLP$(256 \times 3 \times 3, 100)$
BatchNorm	BatchNorm	BatchNorm
ReLU	Convolution$(60 \times 3 \times 3)$	ReLU
MaxPooling(2×2)	BipolarUpsampling(2)	MaxPooling(2×2)
SoftMax(7)		SoftMax(2)

A.2 Deep Q-Learning

Table A.2: Topology for the deep Q-network from Chapter 6. For the convolutional layers: $(filters \times filterWidth \times filterHeight, stride)$.

Encoder	Decoder	Q-Network
Convolution$(32 \times 5 \times 5, 1)$ BatchNorm ReLU-n	Sigmoid BatchNorm Convolution$(1 \times 5 \times 5, 1)$	Convolution$(64 \times 3 \times 3, 1)$ BatchNorm ReLU
Convolution$(64 \times 3 \times 3, 2)$ BatchNorm ReLU-n	ReLU BatchNorm Convolution$(32 \times 3 \times 3, 1)$ BipolarUpsampling(2)	Convolution$(96 \times 3 \times 3, 2)$ BatchNorm ReLU
Convolution$(64 \times 3 \times 3, 1)$ BatchNorm ReLU-n	ReLU BatchNorm Convolution$(64 \times 3 \times 3, 1)$	Convolution$(128 \times 3 \times 3, 2)$ BatchNorm ReLU
		Convolution$(128 \times 3 \times 3, 2)$ BatchNorm ReLU SoftMax(9)

List of References

[1] M. Lewis, J. M. Haviland-Jones, and L. F. Barrett, "Handbook of Emotions.," *Contemporary Sociology*, vol. 24, p. 298, may 1995.

[2] C. Lamm and J. Majdandžić, "The role of shared neural activations, mirror neurons, and morality in empathy - A critical comment," *Neuroscience Research*, vol. 90, pp. 15–24, 2015.

[3] P. Ekman, "Basic emotions," 1999.

[4] D. chiopu, "Using Artificial Neural Networks in a Pattern Recognition Control System.," *Petroleum - Gas University of Ploiesti Bulletin, Technical Series*, vol. 61, no. 3, pp. 365–370, 2009.

[5] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning Internal Representations by Error Propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition* (D. E. Rumelhart, J. L. McClelland, and C. PDP Research Group, eds.), pp. 399–421, MIT Press, 1986.

[6] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural Networks*, vol. 4, pp. 251–257, jan 1991.

[7] B. Csáji, "Approximation with artificial neural networks," *MSc. thesis*, p. 45, 2001.

[8] M. Nielsen, *Neural Networks and Deep Learning*. Determination Press, 2015.

[9] L. Deng and D. Yu, "Deep Learning: Methods and Applications," *Foundations and Trends® in Signal Processing*, vol. 7, no. 3-4, pp. 197—-387, 2013.

[10] Y. LeCun, "Generalization and network design strategies," 1989.

[11] D. H. Hubel and T. N. Wiesel, "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex.," *The Journal of physiology*, vol. 160, pp. 106–54, jan 1962.

[12] T. Brosch and R. Tam, "Efficient Training of Convolutional Deep Belief Networks in the FrequencyDomain for Application to High-Resolution 2Dand 3D Images," *Neural computation*, pp. 211–227, 2015.

[13] P. Burkert, F. Trier, M. Z. Afzal, A. Dengel, and M. Liwicki, "DeXpression: Deep convolutional neural network for expression recognition," *arXiv preprint*, pp. 1–8, 2015.

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *Arxiv.Org*, vol. 7, pp. 171–180, dec 2015.

[15] C. Szegedy, W. Liu, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," tech. rep.

[16] A. Krizhevsky, L. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *NIPS*, pp. 1106–1114, 2012.

[17] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," tech. rep., 2015.

[18] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *International Journal of Computer Vision*, vol. 104, no. 2, pp. 154–171, 2013.

[19] G. Levi and T. Hassner, "Emotion Recognition in the Wild via Convolutional Neural Networks and Mapped Binary Patterns," in *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction - ICMI '15*, (New York, New York, USA), pp. 503–510, ACM Press, 2015.

[20] A. Ruiz-Garcia, M. Elshaw, A. Altahhan, and V. Palade, "Deep learning for emotion recognition in faces," in *Lecture Notes in Computer Science (including*

subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 9887 LNCS, pp. 38–46, Springer, Cham, 2016.

[21] D. Duncan, G. Shine, and C. English, "Facial Emotion Recognition in Schizophrenia ,"

[22] S. Ouellet, "Real-time emotion recognition for gaming using deep convolutional network features," *CoRR*, vol. abs/1408.3, p. 6, aug 2014.

[23] A. Ruiz-Garcia, M. Elshaw, A. Altahhan, and V. Palade, "A hybrid deep learning neural approach for emotion recognition from facial expressions for socially assistive robots," *Neural Computing and Applications*, vol. 29, pp. 359–373, apr 2018.

[24] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, pp. 273–297, sep 1995.

[25] K. S. Burbank, "Mirrored STDP Implements Autoencoder Learning in a Network of Spiking Neurons," *PLoS Computational Biology*, vol. 11, no. 12, pp. 1–26, 2015.

[26] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, *Journal of machine learning research : JMLR.*, vol. 11. MIT Press, 2010.

[27] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, "Adversarial Autoencoders," 2015.

[28] I. Goodfellow, J. Pouget-Abadie, M. M. A. in neural . . . , and U. 2014, "Generative adversarial nets," *Advances in Neural Information Processing Systems 27*, pp. 2672–2680, 2014.

[29] L. Tran, X. Yin, and X. Liu, "Disentangled representation learning GAN for pose-invariant face recognition," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 1283–1292, 2017.

[30] J. Zhao, L. Xiong, K. Jayashree, J. Li, F. Zhao, Z. Wang, S. Pranata, S. Shen, S. Yan, and J. Feng, "Dual-Agent GANs for Photorealistic and Identity Preserving Profile Face Synthesis," *Nips 2017*, no. 15, pp. 1–11, 2017.

[31] J. Chen, J. Konrad, and P. Ishwar, "VGAN-Based Image Representation Learning for Privacy-Preserving Facial Expression Recognition," tech. rep.

[32] S. A. Israel, J. Goldstein, J. S. Klein, J. Talamonti, F. Tanner, S. Zabel, P. A. Sallee, and L. McCoy, "Generative Adversarial Networks for Classification," in *2017 IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*, pp. 1–4, IEEE, oct 2017.

[33] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, no. 2010, pp. 8609–8613, 2013.

[34] Y. Nesterov, "A Method of Solving a Convex Programming Problem with Convergence Rate O(1/2)," *Soviet Mathematics Doklady*, vol. 27, 1983.

[35] M. Riedmiller and H. Braun, "A direct adaptive method for faster backpropagation learning: The RPROP algorithm," in *IEEE International Conference on Neural Networks - Conference Proceedings*, vol. 1993-Janua, pp. 586–591, 1993.

[36] D. P. Kingma and J. L. Ba, "Adam," pp. 1–15, 2015.

[37] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.

[38] A. Krogh and J. A. Hertz, "A Simple Weight Decay Can Improve Generalization," *Advances in Neural Information Processing Systems*, vol. 4, pp. 950–957, 1992.

[39] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," feb 2015.

[40] A. Ruiz-Garcia, M. Elshaw, A. Altahhan, and V. Palade, "Stacked deep convolutional auto-encoders for emotion recognition from facial expressions," in *Proceedings of the International Joint Conference on Neural Networks*, vol. 2017-May, pp. 1586–1593, IEEE, may 2017.

[41] Y. Bengio, "Learning Deep Architectures for AI," *Foundations and Trends R in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.

[42] I. Goodfellow, Bengio, Yoshua, and A. Courville, *Deep Learning*. MIT Press, 2016.

[43] T. Ahsan, T. Jabid, and U.-P. Chong, "Facial Expression Recognition Using Local Transitional Pattern on Gabor Filtered Facial Images," *IETE Technical Review*, vol. 30, no. 1, p. 47, 2013.

[44] F. Z. Chelali and A. Djeradi, "Face Recognition Using MLP and RBF Neural Network with Gabor and Discrete Wavelet Transform Characterization: A Comparative Study," *Mathematical Problems in Engineering*, vol. 2015, pp. 1–16, 2015.

[45] A. Ruiz-Garcia, M. Elshaw, A. Altahhan, and V. Palade, "Emotion Recognition Using Facial Expression Images for a Robotic Companion," in *Engineering Applications of Neural Networks: 17th International Conference, EANN 2016, Aberdeen, UK, September 2-5, 2016, Proceedings*, pp. 79–93, Springer, Cham, 2016.

[46] A. S. M. Sohail and P. Bhattacharya, "Classifying Facial Expressions Using Level Set Method Based Lip Contour Detection and Multi-Class Support Vector Machines," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 25, no. 06, pp. 835–862, 2011.

[47] P. P. Paul, M. M. Monwar, M. L. Gavrilova, and P. S. P. Wang, "Rotation Invariant Multivew Face Detection Using Skin Color Regressive Model and Support Vector Regression," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 24, pp. 1261–1280, dec 2010.

[48] S. A. Khan, A. Hussain, M. Usman, M. Nazir, N. Riaz, and A. M. Mirza, "Robust face recognition using computationally efficient features," *Journal of Intelligent & Fuzzy Systems*, vol. 27, no. 6, pp. 3131–3143, 2014.

[49] R. S. Sutton and A. G. Barto, *Reinforcement learning : an introduction*. MIT Press, 1998.

[50] M. Harandi, M. Ahmadabadi, and B. Araabi, "Face recognition using reinforcement learning," *2004 International Conference on Image Processing, 2004. ICIP '04.*, vol. 4, no. January, pp. 2709–2712, 2004.

[51] B. Goodrich and I. Arel, "Reinforcement Learning based Visual Attention with Application to Face Detection," pp. 19–24, 2012.

[52] P. Wang, W.-H. Lin, K.-M. Chao, and C.-C. Lo, "A Face-Recognition Approach Using Deep Reinforcement Learning Approach for User Authentication," in *2017 IEEE 14th International Conference on e-Business Engineering (ICEBE)*, pp. 183–188, IEEE, nov 2017.

[53] Y. Rao, J. Lu, and J. Zhou, "Attention-aware Deep Reinforcement Learning for Video Face Recognition," *Int. Conf. on Computer Vision (ICCV)*, pp. 3931–3940, 2017.

[54] N. Jain, A. Kumar, P. Shamsolmoali, and M. Zareapoor, "Hybrid deep neural networks for face emotion recognition," *Pattern Recognition Letters*, vol. 115, pp. 101–106, nov 2018.

[55] Y. Rao, J. Lu, and J. Zhou, "Attention-aware Deep Reinforcement Learning for Video Face Recognition," tech. rep.

[56] Z. Yu and C. Zhang, "Image based Static Facial Expression Recognition with Multiple Deep Network Learning," in *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction - ICMI '15*, (New York, New York, USA), pp. 435–442, ACM Press, 2015.

[57] A. Mollahosseini, D. Chan, and M. H. Mahoor, "Going Deeper in Facial Expression Recognition using Deep Neural Networks," *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1–10, mar 2015.

[58] G. Castellano, I. Leite, A. Pereira, C. Martinho, A. Paiva, and P. W. Mcowan, "Multimodal Affect Modeling and Recognition for Empathic Robot Companions," *International Journal of Humanoid Robotics*, vol. 10, no. 01, p. 1350010, 2013.

[59] S. Kumano, K. Otsuka, J. Yamato, E. Maeda, and Y. Sato, "Pose-invariant facial expression recognition using variable-intensity templates," *International Journal of Computer Vision*, vol. 83, no. 2, pp. 178–194, 2009.

[60] B. Aksasse, H. Ouanan, and M. Ouanan, "Novel approach to pose invariant face recognition," *Procedia Computer Science*, vol. 110, pp. 434–439, jan 2017.

[61] M. Kan, S. Shan, H. Chang, and X. Chen, "Stacked progressive auto-encoders (SPAE) for face recognition across poses," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1883–1890, IEEE, jun 2014.

[62] D. Lundqvist, A. Flykt, and A. Öhman, "The Karolinska Directed Emotional Faces - KDEF CD ROM from Department of Clinical Neuroscience, Psycology section," *Karolinska Institutet*, pp. 3–5, 1998.

[63] X. Zhu and D. Ramanan, "Face detection, pose estimation, and landmark localization in the wild," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2879–2886, IEEE, jun 2012.

[64] D. Hamester, P. Barros, and S. Wermter, "Face expression recognition with a 2-channel Convolutional Neural Network," in *Proceedings of the International Joint Conference on Neural Networks*, vol. 2015-Septe, pp. 1–8, IEEE, jul 2015.

[65] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker, "Multi-PIE," in *2008 8th IEEE International Conference on Automatic Face & Gesture Recognition*, pp. 1–8, IEEE, sep 2008.

[66] A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman, "From few to many: Illumination cone models for face recognition under variable lighting and pose," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, pp. 643–660, jun 2001.

[67] M. Lyons, S. Akamatsu, M. Kamachi, and J. Gyoba, "Coding facial expressions with Gabor wavelets," in *Proceedings - 3rd IEEE International Conference on Automatic Face and Gesture Recognition, FG 1998*, pp. 200–205, 1998.

[68] F. Wallhoff, B. Schuller, M. Hawellek, and G. Rigoll, "Efficient recognition of authentic dynamic facial expressions on the feedtum database," in *2006 IEEE International Conference on Multimedia and Expo, ICME 2006 - Proceedings*, vol. 2006, pp. 493–496, IEEE, jul 2006.

[69] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," *PMLR*, vol. 9, pp. 249–256, 2010.

[70] A. Krizhevsky and G. Hinton, "Convolutional deep belief networks on cifar-10," *Unpublished manuscript*, pp. 1–9, 2010.

[71] A. Ruiz-Garcia, V. Palade, M. Elshaw, and I. Almakky, "Deep Learning for Illumination Invariant Facial Expression Recognition," in *Proceedings of the International Joint Conference on Neural Networks*, (Rio de Janeiro), IEEE, 2018.

[72] P. Vincent, and H. Larochelle, "Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion Pierre-Antoine Manzagol," *Journal of Machine Learning Research*, vol. 11, pp. 3371–3408, 2010.

[73] D. H. Liu, K. M. Lam, and L. S. Shen, "Illumination invariant face recognition," *Pattern Recognition*, vol. 38, pp. 1705–1716, oct 2005.

[74] C. Tosik, A. Eleyan, and M. Salman, "Illumination invariant face recognition system," in *2013 21st Signal Processing and Communications Applications Conference, SIU 2013*, pp. 1–4, IEEE, apr 2013.

[75] X. Chen, X. Lan, G. Liang, J. Liu, and N. Zheng, "Pose-and-illumination-invariant face representation via a triplet-loss trained deep reconstruction model," *Multimedia Tools and Applications*, vol. 76, pp. 22043–22058, nov 2017.

[76] O. Gupta, D. Raviv, and R. Raskar, "Deep video gesture recognition using illumination invariants," *Arxiv*, pp. 1–9, 2016.

[77] Ekman and W. Friesen, "Facial Action Coding System: A Technique for the Measurement of Facial Movement.," *Consulting Psychologists Press, Palo Alto.*, 1978.

[78] A. Ruiz-Garcia, N. Webb, V. Palade, M. Eastwood, and M. Elshaw, "Deep Learning for Real Time Facial Expression Recognition in Social Robots," in *Proceedings of the International Conference on Neural Information Processing*, 2018.

[79] W. Shi, J. Caballero, F. Huszar, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1874–1883, 2016.

[80] M. Lin, Q. Chen, and S. Yan, "Network In Network," *arXiv preprint*, p. 10, dec 2013.

[81] N. M. Hewahi and A. R. M. Baraka, "Impact of Ethnic Group on Human Emotion Recognition Using Backpropagation Neural Network," *BRAIN. Broad Research in Artificial*, pp. 20–27, 2012.

[82] P. Viola and M. Jones, "Robust real-time object detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, 2001.

[83] A. Jourabloo, M. Ye, X. Liu, and L. Ren, "Pose-Invariant Face Alignment with a Single CNN," in *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2017-Octob, pp. 3219–3228, jul 2017.

[84] C. Shu, X. Ding, and C. Fang, "Histogram of the oriented gradient for face recognition," *Tsinghua Science and Technology*, vol. 16, pp. 216–224, apr 2011.

[85] O. Jesorsky, K. J. Kirchberg, and R. W. Frischholz, "Robust Face Detection Using the Hausdorff Distance," *In Proc. Third International Conference on Audio- and Video-based Biometric Person Authentication*, no. June, pp. 90–95, 2001.

[86] R. Keys, "Cubic convolution interpolation for digital image processing: IEEE Trans. Acoust., Speech," *Signal Process., ASSP-29*, no. 6, p. 1153, 1981.

[87] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and

D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[88] V. Mnih, D. Silver, and M. Riedmiller, "Dqn," *Nips*, pp. 1–9, 2013.

[89] H. Jiang and E. Learned-Miller, "Face Detection with the Faster R-CNN," in *Proceedings - 12th IEEE International Conference on Automatic Face and Gesture Recognition, FG 2017 - 1st International Workshop on Adaptive Shot Learning for Gesture Understanding and Production, ASL4GUP 2017, Biometrics in the Wild, Bwild 2017, Heteroge*, pp. 650–657, 2017.

[90] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2017.

[91] P. Harár, R. Burget, and M. K. Dutta, "Speech Emotion Recognition with Deep Learning," *Signal Processing and Integrated Networks (SPIN)*, pp. 4–7, feb 2017.

[92] K. Schindler, L. Van Gool, and B. de Gelder, "Recognizing emotions expressed by body pose: A biologically inspired neural model," *Neural Networks*, vol. 21, no. 9, pp. 1238–1246, 2008.

[93] Z. Wen, R. Xu, and J. Du, "A novel convolutional neural networks for emotion recognition based on EEG signal," in *2017 International Conference on Security, Pattern Analysis, and Cybernetics (SPAC)*, pp. 672–677, IEEE, dec 2017.

[94] R. Collobert, K. Kavukcuoglu, and C. Farabet, "Torch7: A matlab-like environment for machine learning," tech. rep., 2011.

[95] Y. Huang and S. M. Khan, "DyadGAN: Generating Facial Expressions in Dyadic Interactions," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, vol. 2017-July, pp. 2259–2266, IEEE, jul 2017.

[96] Y. Zhou and B. E. Shi, "Photorealistic facial expression synthesis by the conditional difference adversarial autoencoder," in *2017 7th International Conference on Affective Computing and Intelligent Interaction, ACII 2017*, vol. 2018-Janua, pp. 370–376, 2018.

[97] F. Qiao, N. Yao, Z. Jiao, Z. Li, H. Chen, and H. Wang, "Geometry-Contrastive Generative Adversarial Network for Facial Expression Synthesis," 2018.